1991年7月1日発行（毎月1回1日発行）第10巻7号通巻111号　昭和58年11月2日第三種郵便物認可

PERSONAL COMPUTER MAGAZINE for MZ, X1, and X68000

# Oh!X

## 特集 パーソナルツール,BASIC

グラフィックパッケージMAGIC用外部関数/カットファイル作成
X1用買い物ゲーム The Master of Payment/シューティング68K
tinyCalcの応用/マイコンショウ&ビジネスショウレポート
別冊付録 X-BASICポケットリファレンスブック

7
1991

SOFT BANK オー！エックス
特別定価600円

SHARP

仕事だけのパソコンや
ワープロみたいなパソコンは、
いらない。

# 父のパソコンを超えろ。

**シャープX68000パソコン教室開催中**
- 会場：四谷教室
- コース：入門コース・表集計コース・音楽コース・絵画コース
- 申込受付電話番号(03)3260-8365
- 受講料：2,000円(税別)

夢、創ります。

## 「第1回全日本X68000芸術祭」開催

X68000XVIデビューを記念して、オリジナルソフトウェア・作品コンテストを開催いたします。7月からの地区予選に始まる全国規模の大会で、日頃の腕試しをするのには絶好の機会。ゲーム、ミュージック、グラフィックス等の各部門へぜひ力作をお寄せください。あなたの自信作が全国のパソコンユーザーの羨望の的になるかもしれません。乞う御期待！

※詳細はX68000店頭でポスター・チラシをご覧ください。

資料請求券 X68000 Oh!X 7係

いまクロック16MHzの俊才、「エクシヴィ」のデビューで5年に及ぶ68000CPUへの探求は、ひとつの結論を得ようとしています。極めたといえば言い過ぎでしょうが、事の深淵に迫ろうと努力するもののみに与えられる深い充足を、私たちスタッフは、これまでX68000を支えていただいたユーザー、ソフトハウス、ハードベンダー諸兄とともに味わいたい心境です。徹底したこだわりと、それを裏付けるアドバンストテクノロジー、世間の逆風を揚力にしてしまう、それなりの魅力と知性を背景として備えたX68000が、パーソナルコンピュータに新しいジャンルを切り拓いてきた歩みは、ご存じの通りです。現在のマルチメディア環境を開発当初から想定していた先見性。一言でいえばクリエイティブマインドということでしょうが、そのグラフィックアビリティ、映像統合コンセプト、サンプリング音源、ウィンドウ環境、そうした単に、とはいえ凄いスペックさえ超えたところにX68000の付加価値は存在します。アプリケーションを走らせるだけのブラックボックス化した、あるいは文房具としてのマシン、それはそれで異論はないのですが、本来的にパーソナルコンピュータがもつ可能性を育む、いわば創造性という観点から物足りなさを覚えることも事実です。X68000は、ある意味ではたいへんな異端児かも知れません。しかし世間から見たその"異能"は、私たちが考えるパーソナルコンピュータとしてはまさにスタンダードに他なりません。いつも新鮮な感動がある、驚きがある。新しい発見がある。"センス"の違いはスペックをも超えて使う人に訴えかける、敢えて68000CPUに執着してきた理由もここにあります。ワークステーションとしての成熟、先見性、創造性の具現化、ユーザーインターフェイスの追求。X68000の進化の過程はここに凝縮されています。

——————————新しい「エクシヴィ」がこのコンセプトをどう発展させたか、ご体感ください。

# 瞬速16MHz、エクシヴィ登場。

**16MHzクロック68000搭載**：OSの高速化、アートワークをパワフルにサポートするクロック周波数16MHzの68000CPUを搭載。クリエイティブワークステーションにふさわしいシステムパフォーマンスを実現しました。

**SX-WINDOW ver1.1搭載**：CPUのクロックアップと合わせ、大幅な処理速度の向上を実現。操作性を一段と高めたニューバージョンです。多機能・高速の強力エディタを搭載。文字選択・外字作成ツールも装備して、スムーズな日本語入力環境をサポート。またプリンタデバイスドライバを搭載し、多彩な印字指定が可能です。もちろん、こうした新しい環境がすべてのX68000で享受できることは言うまでもありません。そして待望のウィンドウアプリケーションもリリースされはじめています。

**高密度メモリ拡張環境**：メインメモリは標準で2Mバイト、本体内部のメイン基板上に6Mバイト増設でき、I/Oスロットを使用せず最大8Mバイトの高速メモリアクセスを実現。さらにI/Oスロットへの増設を含め最大12Mバイトまで拡張できます。数値演算プロセッサも本体内に取り付けられます。

※2MB増設メモリ（ボード型）CZ-6BE2A　標準価格59,800円（税別）、2MB増設メモリ（チップ型）CZ-6BE2B　標準価格54,800円（税別）、数値演算プロセッサ（チップ型）CZ-6BP2　標準価格45,800円（税別）を使用。（すべて別売）

●大容量メディア対応、世界標準SCSIインターフェイス標準装備●X68000シリーズとフルコンパチブル設計●高品位なチタンブラックのニューデザインマンハッタンシェイプ●81MバイトSCSI仕様HDD搭載（CZ-644C）/内蔵可能（CZ-634C）●1024×1024ドットの実画面エリアを装備した高解像度表示（最大表示エリア768×512ドット・65,536色中16色表示）、65,536色同時表示（512×512ドット時）、先駆の高解像度自然色グラフィック●AD PCM、ステレオ8オクターブ8重和音FM音源搭載●オートロード・オートイジェクトの1Mバイト5インチFDD2基搭載●マウス・トラックボール標準装備

X68000
PERSONAL WORKSTATION
XVI
エクシヴィ

X68エクシヴィ
▼
16MHz
新登場

●写真はCZ-644C-TNとCZ-613D-TN。

本体＋キーボード＋マウス・トラックボール
CZ-634C-TN（チタンブラック）　標準価格368,000円（税別）
81MB HDタイプ　CZ-644C-TN（チタンブラック）　標準価格518,000円（税別）

**SUPER** 本体＋キーボード＋マウス・トラックボール
CZ-604C-TN（チタンブラック）　標準価格348,000円（税別）
81MB HDタイプCZ-623C-TN（チタンブラック）　標準価格498,000円（税別）

**PROII** 本体＋キーボード＋マウス
CZ-653C-BK（ブラック）・-GY（グレー）　標準価格285,000円（税別）
40MB HDタイプCZ-663C-BK（ブラック）・-GY（グレー）　標準価格395,000円（税別）

| | | |
|---|---|---|
| ●15型カラーディスプレイテレビ（ドットピッチ0.39mm） | CZ-602D-BK（ブラック）・-GY（グレー※） | 標準価格99,800円（チルトスタンド同梱・税別） |
| ●15型カラーディスプレイテレビ（ドットピッチ0.39mm） | CZ-605D-BK（ブラック）・-GY（グレー） | 標準価格115,000円（スピーカー2個/チルトスタンド同梱・税別） |
| ●15型カラーディスプレイテレビ（ドットピッチ0.31mm） | CZ-613D-TN（チタンブラック）・-BK（ブラック）・-GY（グレー） | 標準価格135,000円（スピーカー2個/チルトスタンド同梱・税別） |
| ●14型カラーディスプレイ（ドットピッチ0.31mm） | CZ-603D-BK（ブラック）・-GY（グレー） | 標準価格84,800円（チルトスタンド同梱・税別） |
| ●14型カラーディスプレイ（ドットピッチ0.31mm） | CZ-604D-BK（ブラック）・-GY（グレー） | 標準価格94,800円（スピーカー2個/チルトスタンド同梱・税別） |
| ●14型カラーディスプレイ（ドットピッチ0.31mm） | CZ-606D-TN（チタンブラック）・-BK（ブラック）・-GY（グレー） | 標準価格79,800円（チルトスタンド同梱・税別） |
| ●21型カラーディスプレイ（ドットピッチ0.52mm） | CU-21HD-BK（ブラック） | 標準価格148,000円（スピーカー2個同梱・税別） |

※印の商品は在庫僅少です。

## 68買ったらEXEクラブに入ろう！

本体同梱の入会申込ハガキを送るだけで、無料入会。3つのメリット！
**メリット1**：会員No.入りオリジナル**会員証電卓**がもらえる。
**メリット2**：各種フェアご優待・イベントご案内等、数々の特典あり。
**メリット3**：X68000の活用情報が手に入る「**EXEおみこし活動**」に参加できる。
※「申込ハガキをなくしてしまった」という方は、「おみこし活動隊」☎(06)886-0354までお電話ください。

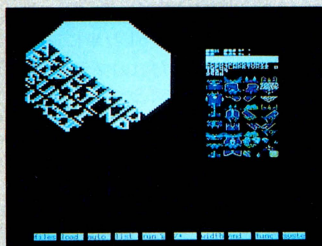
**EXEおみこし活動とは?**
コミュニケーションペーパー「おみこしPRESS」を通じて会員同士が情報を交換、どこまでもX68000を使いこなして盛り上がってしまおう！というのが、その目的。68へのラブコール、会員独自のテクニック・活用法など、あなたの68自慢を「おみこし活動隊」までどうぞ。会員メッセージは随時「おみこしPRESS」に掲載します。

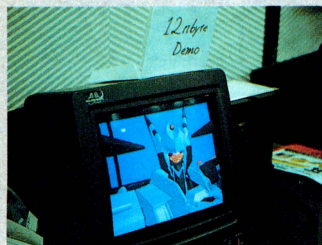
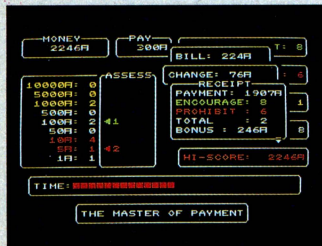
●お問い合わせは…
**シャープ株式会社**
電子機器事業本部システム機器営業部
〒545 大阪市阿倍野区長池町22番22号☎(06)621-1221（大代表）
電子機器事業本部液晶映像システム事業部第2商品企画部
〒162 東京都新宿区市谷八幡町８番地☎(03)3260-1161（大代表）

# Oh!X

## CONT




〈スタッフ〉
●編集長／前田 徹 ●副編集長／植木章夫 ●編集／岡崎栄子 浅井研二 山田純二 ●協力／有田隆也 中森 章 林 一樹 吉田幸一 華門真人 毛利俊行 吉田賢司 影山裕昭 古村 聡 村田敏幸 丹 明彦 三沢和彦 長沢淳博 宮島 靖 金子俊一 浦川博之 石上達也 ●カメラ／杉山和美 ●イラスト／永沢しげる 山田晴久 小栗由香 ●アートディレクター／島村勝頼 ●レイアウト／元木昌子 ADGREEN ●校正／グループこじら


特集 Personal Tool, BASIC

シャープパソコンフォーラム'91

The Master of Payment

ファランクス

A列車で行こうIII

シューティング68K

表紙絵：塚田　哲也

1991 JUL.
7

# E　N　T　S





■広告目次

お望みのパワーシステムへ。

シャープペリフェラルファミリー
X68000

PRO II

## ボード

### 拡張メモリ

NEW
2MB増設RAMボード
(CZ-634C/644C専用)
CZ-6BE2A
標準価格59,800円(税別)
※2MB増設RAM(CZ-6BE2B)専用ソケットを2個用意しています。

NEW
2MB増設RAM
(CZ-634C/644C専用)
CZ-6BE2B
標準価格54,800円(税別)
※本増設RAM(CZ-6BE2B)は、2MB増設RAMボードが必要です。CZ-6BE2A上の専用ソケット(2個用意)に装着ください。
※取付に関してはシャープお客様ご相談窓口にてご相談ください。

1MB増設RAMボード
(CZ-600C専用)
★CZ-6BE1
標準価格35,000円(税別)

1MB増設RAMボード
(CZ-601C/611C/652C/653C/662C/663C用)
CZ-6BE1B
標準価格28,000円(税別)

2MB増設RAMボード※6
CZ-6BE2
標準価格79,800円(税別)

4MB増設RAMボード※6
CZ-6BE4
標準価格138,000円(税別)

### インターフェイス

NEW
SCSIボード※7
CZ-6BS1
標準価格29,800円(税別)
(ソフトウェア(SCSIユーティリティ)同梱)

ユニバーサルI/Oボード
★CZ-6BU1
標準価格39,800円(税別)

GP-IBボード
★CZ-6BG1
標準価格59,800円(税別)

増設用RS-232Cボード
(2チャンネル)
★CZ-6BF1
標準価格49,800円(税別)

### MIDI

MIDIボード
CZ-6BM1
標準価格26,800円(税別)

### FAX

FAXボード
CZ-6BC1
標準価格79,800円(税別)

### 数値演算プロセッサ

数値演算プロセッサボード
CZ-6BP1
標準価格79,800円(税別)

NEW
数値演算プロセッサ
(CZ-634C/644C専用)
CZ-6BP2
標準価格45,800円(税別)
※取付に関してはシャープお客様ご相談窓口にてご相談ください。
※特別ケース入りです。

## ネットワーク

### モデム

モデムユニット※8
CZ-8TM2
標準価格49,800円(税別)
(RS-232Cケーブル同梱)

### RS-232Cケーブル

RS-232Cケーブル
(平行接続型)
CZ-8LM1
標準価格7,200円(税別)

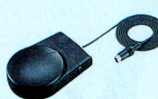
RS-232Cケーブル
(クロス接続型)
CZ-8LM2
標準価格7,200円(税別)

### LANボード

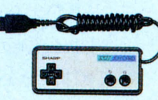
LANボード
CZ-6BL1
標準価格268,000円(税別)
(イーサネット用)

CZ-6BL2
標準価格298,000円(税別)
(イーサネット/チーパネット両用)
※電源ユニット・ソフトウェア
(ネットワークドライバVer1.0)同梱

## 入力

インテリジェントコントローラ
CZ-8NJ2
標準価格23,800円(税別)

マウス・トラックボール
CZ-8NM3
標準価格9,800円(税別)

トラックボール
CZ-8NT1
標準価格13,800円(税別)

マウス
CZ-8NM2A
標準価格6,800円(税別)

ジョイカード
CZ-8NJ1
標準価格1,700円(税別)

## その他

### 拡張スロット

拡張I/Oボックス(4スロット)
(CZ-600C/601C/602C/603C/604C/611C/612C/613C/623C/634C/644C用)
CZ-6EB1-BK
★CZ-6EB1
標準価格88,000円(税別)

### スピーカー

アンプ内蔵
スピーカーシステム(2本1組)
AN-S100
標準価格36,600円(税別)

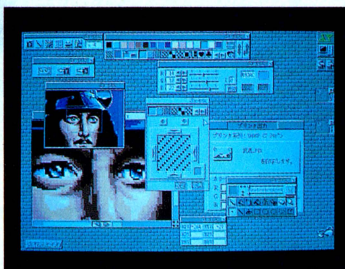
### システムラック

システムラック
(CZ-600C/601C/602C/603C/604C/611C/612C/613C/623C/634C/644C用)
CZ-6SD1
標準価格44,800円(税別)

★印の商品は在庫僅少です。

■製品改良のため仕様の一部を予告なく変更することがあります。またこの広告の色調は印刷のため実物とは多少異なる場合もありますのであらかじめご了承ください。

CZ-600C用)、CZ-6BE1B 標準価格28,000円(税別・CZ-601C、CZ-611C、652C、653C、662C、663C用)を増設してください。 ※7 CZ-600C、601C、602C、603C、611C、612C、613Cに装着の場合、I/Oスロット2に装着ください。CZ-652C、653C、662C、663Cに装着の場合はI/Oスロット4に装着ください。また、CZ-6BG1、6BU1、6BL1、6BL2、6BN1などのボードは、接続コネクタとの関係で本ボードとの併用はできませんのでご注意ください。なお、本ボードはX68000用OS Human 68K ver.2.0以上にてご使用ください。 ※8 モデムユニットCZ-8TM2に同梱のソフトはX1/X1ターボシリーズ用です。

電子機器事業本部液晶映像システム事業部第2商品企画部 〒162 東京都新宿区市谷八幡町8番地 ☎(03)3260-1161(大代表)

X68000 APPLICATION REVIEW

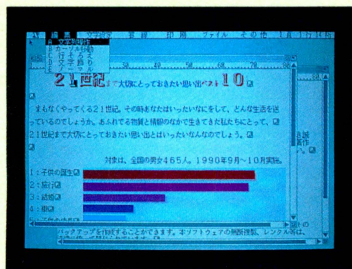
# NEW RELEASE

## マルチワープロ Multiword PRO-68K

ウィンドウでWYSIWYG編集。
カラーグラフィック、高速テキストモード。

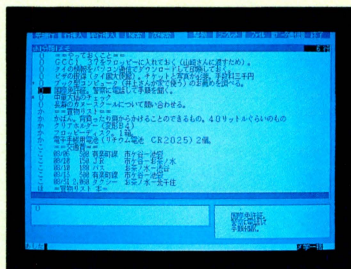
CZ-225BS　標準価格32,000円(税別)

WYSIWYGな編集が行えるウィンドウモードと素早い編集が行えるテキストモードをサポート。グラフィックを文章中に自由にレイアウトできます。また同一文章での複数の改行幅指定を可能にするなど多彩な機能を装備。レーザープリンタ、カラー印字(8色)の高品位プリントアウトも可能です。

※メインメモリ2MB必要です。

## Teleportion PRO-68K

パソコン通信も、エディタも。
【メモリ常駐型】の優れもの。

CZ-258BS　標準価格22,800円(税別)

他のソフトウェアを実行中でも任意に呼び出して使える メモリ常駐型 のソフトウェアです。パソコン通信/エディタ/カレンダー/スケジュール/住所録/メモ帳/関数電卓の機能を文具感覚でお使いいただけます。「シャープ電子手帳」のデータをX68000で簡単に入力・編集することができます。

※メインメモリ2MB必要です。※StationeryPRO-68K(CZ-240BS)をお持ちの方には有償バージョンアップサービスを行います。

## NEW PrintShop ver2.0 PRO-68K

処理速度の高速化を実現。
Zeit日本語ベクトルフォントをサポート。

CZ-265HS　7月発売予定

オリジナリティを活かせるホームプロダクティビリティツールです。処理速度の高速化を実現。カセットレーベル、カレンダー作成に対応したほか、モノクロデータの編集などグラフィックエディタを強化し高機能テキストエディタを内蔵しました。Zeit日本語ベクトルフォントも使用可能です。

※メインメモリ2MB必要です。※NEW Print Shop PRO-68K(CZ-221HS)をお持ちの方には有償バージョンアップサービスを行います。

●お問い合わせは…シャープ㈱電子機器事業本部液晶映像システム事業部第2商品企画部 〒162 東京都新宿区市谷八幡町8番地 ☎(03)3260-1161(大代表)へ。 シャープ株式会社

SHARP

X68000 XVIデビュー記念キャンペーン

「夢、創ります。山下章氏プロデュース」

# 第1回全日本X68000
# 芸術祭開催

X68000アイドル山下章氏司会、進行による
ユーザー参加型作品コンテスト

■主催：シャープ株式会社 電子機器事業本部 システム機器営業部
■共催：シャープエレクトロニクス販売株式会社各統轄営業部
東京中央シャープ販売㈱・浪速シャープ電機㈱・
沖縄シャープ電機㈱
■協賛：出版社・ソフトハウス・サードパーティ・主要販売店

7月からの地区予選に始まる全国規模のオリジナルソフトウェア・作品コンテストです。日頃の腕試しに、ぜひ力作をお寄せください。全国のパソコンユーザーを魅了する芸術祭グランプリをめざして、あなたの自信作はどこまで勝ち進めるか？乞う御期待！

作品募集中！

〈作品応募要項〉■**作品基準**：パーソナルコンピュータ（メーカー、機種を問わず）で制作した、オリジナル未発表のプログラム、グラフィックス、コンピュータ・ミュージック等であること。なお応募作品（制作に使用したアプリケーション・ソフト等以外の部分）の著作権は、すべてシャープ㈱に帰属します。■**部門**：①**ゲーム部門**、②**ミュージック部門**（自作の曲／一般曲・ゲームミュージックのアレンジ等、MIDI使用も可。）③**グラフィックス部門**（Z'sSTAFF PRO-68K、DOGA等のツールを使用して描いたものなど画面上に表示されるグラフィックスなら何でも可。）④**その他部門**：（ユーティリティ／一発ギャグ／パフォーマンス／ビジネス利用／その他）※応募は、1部門につき1人1作。1人複数部門応募は可。また団体制作も可。■**応募資格**：各予選ブロックの地域の住人であること。■**応募方法**：プログラム・ディスクに住所／氏名／年齢／職業（学校名・学年）／電話番号／開発に要した期間／開発に使用・利用したツール名／セールスポイント／取り扱い上の注意／動作に必要とする特殊機材を添え、各地区の応募先まで郵送してください。締め切りはその地区の地区大会開催日の2週間前（必着）です。■**賞・賞品**：〈地区予選〉●各地区大会大賞（1点）トロフィー、賞状、副賞●入選（首都圏3点、近畿2点、中部・九州1点、他地区なし）賞状、副賞●協賛各社賞・賞状、副賞〈全国大会〉●第1回全日本X68000芸術祭グランプリ（1点）トロフィー、賞状、副賞：光磁気ディスクユニット（CZ-6M01）ペアでの海外旅行（旅行クーポン）●ゲーム・ミュージック・グラフィック等各部門賞・賞状、副賞●協賛各社賞・賞状、副賞

※詳細は店頭のチラシをご覧ください。

| | 開催地 | 開催日 | 会場 | 入選枠 | 対象都道府県 | 応募・問い合わせ先 | 締切日 |
|---|---|---|---|---|---|---|---|
| 7月 | 四国（高松） | 7月21日㈰ | シャープ高松ビル 5Fイベントホール 高松市朝日町6-2-8 ☎0878-23-4860 | 大賞1点 | 徳島・香川・愛媛・高知 | 〒760 高松市朝日町6-2-8 シャープエレクトロニクス販売㈱四国統轄（営）パソコン担当、辻井部長・細川係長 ☎0878-23-4860㈹ | 7月5日㈮ |
| 8月 | 北海道（札幌） | 8月4日㈰ | シャープ札幌ビル 4Fホール 札幌市西区二十四軒1条7-3-17 ☎011-642-8111 | 大賞1点 | 北海道 | 〒063 札幌市西区二十四軒1条7-3-17 シャープエレクトロニクス販売㈱ 北海道統轄（営） パソコン担当、長谷田担当 ☎011-642-8111㈹ | 7月26日㈮ |
| 9月 | 東北（仙台） | 9月8日㈰ | シャープ仙台ビル 4Fホール 仙台市若林区卸町東3-1-27 ☎022-288-9111 | 大賞1点 | 青森・山形・岩手・福島・宮城・秋田 | 〒983 仙台市若林区卸町東3-1-27 シャープエレクトロニクス販売㈱ 東北統轄（営） パソコン担当、岡本部長・阿部課長 ☎022-288-9111㈹ | 8月23日㈮ |
| 9月 | 中国（広島） | 9月15日㈰ | 広島市西区民センター 3F大会議室A 広島市西区横川新町6-1 ☎082-234-1960 | 大賞1点 | 鳥取・島根・岡山・広島・山口 | 〒731-01 広島市安佐南区西原2-13-4 シャープエレクトロニクス販売㈱ 中国統轄（営） パソコン担当、青木部長・石井担当 ☎082-874-2282㈹ | 8月30日㈮ |
| 9月 | 北関東（宇都宮） | 9月22日㈰ | 護国会館 平安の間 宇都宮市陽西町1-37 ☎0286-22-3180 | 大賞1点 | 茨城・群馬・栃木 | 〒320 宇都宮市不動前4-2-41 シャープエレクトロニクス販売㈱ 北関東統轄（営） パソコン担当、岩田部長・川俣係長 ☎0286-35-1151㈹ | 9月6日㈮ |
| 10月 | 神奈川（横浜） | 10月6日㈰ | 神奈川県労働総合センター 5F大講堂 横浜市磯子区中原1-1-28 ☎045-773-2250 | 大賞1点 | 神奈川 | 〒235 横浜市磯子区中原1-2-23 シャープエレクトロニクス販売㈱ 神奈川統轄（営） パソコン担当、常次部長 ☎045-753-5501㈹ | 9月20日㈮ |
| 10月 | 中部（名古屋） | 10月20日㈰ | シャープ名古屋ビル 7Fホール 名古屋市中川区山王3-5-5 ☎052-323-5111 | 大賞1点 入選1点 | 静岡・愛知・長野・岐阜・三重 | 〒454 名古屋市中川区山王3-5-5 シャープエレクトロニクス販売㈱ 中部統轄（営） パソコン担当、山口課長 ☎052-323-5111㈹ | 10月4日㈮ |
| 11月 | 北陸（金沢） | 11月3日㈰ | 労済会館 金沢市西念1-12-22 ☎0762-23-5911 | 大賞1点 | 富山・石川・福井 | 〒921 石川県石川郡野々市町字御経塚町1096-1 シャープエレクトロニクス販売㈱北陸統轄（営） パソコン担当、小林担当 ☎0762-49-1181㈹ | 10月18日㈮ |
| 11月 | 近畿（大阪） | 11月10日㈰ | ビジネスセンター三水 5Fホール 大阪市浪速区日本橋東3-14-10 ☎06-632-5141 | 大賞1点 入選2点 | 滋賀・京都・大阪・兵庫・奈良・和歌山 | 〒556 大阪市浪速区恵美須西1-2-9 シャープエレクトロニクス販売㈱ 近畿統轄（営） パソコン担当、岡本課長・細川係長 ☎06-631-1181㈹ | 10月25日㈮ |
| 11月 | 首都圏（東京） | 11月24日㈰ | シャープ東京支社 8Fエルムホール 東京都新宿区市ヶ谷八幡町8 ☎03-3260-1161 | 大賞1点 入選3点 | 埼玉・山梨・千葉・新潟・東京 | 〒162 東京都新宿区市ヶ谷八幡町8 シャープエレクトロニクス販売㈱ 首都圏統轄（営） パソコン営業部、福井部長・前田課長 ☎03-3266-8248 | 11月8日㈮ |
| 12月 | 九州（福岡） | 12月14日㈯ | KC会館 2F大ホール 福岡市博多区博多駅前3-4-2 ☎092-451-5971 | 大賞1点 入選1点 | 福岡・佐賀・長崎・熊本・大分・宮崎・鹿児島・沖縄 | 〒816 福岡市博多区井相田2-12-1 シャープエレクトロニクス販売㈱ 九州統轄（営） パソコン営業部、北山部長・岩崎課長 ☎092-501-6806 | 11月29日㈮ |
| 2月 | 補選（大阪） | 2月9日㈰（予定） | （未定） | 入選2点 | 全国 | 〒545 大阪市阿倍野区長池町22-22 シャープ株式会社 電子機器事業本部 システム機器営業部 ☎06-621-1221㈹ | 1月24日㈮（予定） |

協賛雑誌（50音順）：I/O（工学社）・アスキー（アスキー）・Oh!X（ソフトバンク）・コンプティーク（角川書店）・POPCOM（小学館）・マイコン（電波新聞社）・マイコンBASICマガジン（電波新聞社）・LOGIN（アスキー）

## 68買ったらEXEクラブに入ろう！

本体同梱の入会申込ハガキを送るだけで、無料入会。3つのメリット！
**メリット1**：会員No.入りオリジナル**会員証電卓**がもらえる。
**メリット2**：各種フェアご優待・イベントご案内等、数々の特典あり。
**メリット3**：X68000の活用情報が手に入る「**EXEおみこし活動**」に参加できる。
※「申込ハガキををなくしてしまった」という方は、右記「おみこし活動隊」までお電話ください。

**EXEおみこし活動とは？**

コミュニケーションペーパー「おみこしPRESS」を通じて会員同士が情報を交換、どこまでもX68000を使いこなして盛り上がってしまおう！というのが、その目的。68へのラブコール、会員独自のテクニック・活用法など、あなたの68自慢を「おみこし活動隊」までどうぞ。会員メッセージは随時「おみこしPRESS」に掲載します。

さらに熱心な会員のために、「**おみこしかつぎ人**」制度も設けました。「かつぎ人」3つのメリットは…❶X68000情報交換会「**おみこしかつぎ人の集い**」に参加できる。❷68最新ソフト・各周辺機器が一覧できる「**ソフトウェア・フィールド**」を半年1回送付。❸「**おみこしPRESS**」毎号送付。「かつぎ人」になれば68ユーザーとして一層充実すること間違いなしです。

●「おみこしかつぎ人」になるには、年会費（おみこしかつぎ代）が必要です。個人入会3,000円／グループ入会（5人1組）2,500円・郵便振込にて申込受付。●詳細は店頭の「おみこしPRESS」をご覧になるか、または「おみこし活動隊」にお電話ください。
**おみこし活動隊☎(06)886-0354**

●お問い合わせは…シャープ㈱電子機器事業本部システム機器営業部 〒545 大阪市阿倍野区長池町22番22号 ☎(06)621-1221(大代表) シャープ株式会社

# Mu-1 Musicstudio Super

[ミューワン スーパー]

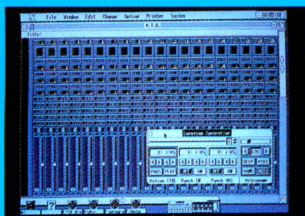
近日発売 ¥39,800(税別)

## “Musicstudio”スペシャルバージョン新登場!

Mu-1 Superは、マルチレコーダー“Musicstudio PRO-68K Ver2.0”にMu-1のミュージ郎コンバート機能、グラフィック機能を合体させ、さらにパワーアップしたスペシャルバージョンです。

### ◆特 長◆

❶ミュージ郎(98用5"2HD)データコンバートはもとより、マックやアタリなど世界のMIDIソフトと双方向のデータコンバートが可能な“スタンダードMIDIファイル”対応 ❷Sound Canvas(ローランド社新音源SC-55)のGS規格に対応した新デザイン“MTRウィンドウ” ❸CM-32L、64、MT-32の音色を自由に作り変えられる“LA音源音色エディタ搭載 ❹ドラム音色変更可能な52チャンネルミキサー搭載(CM-32L、64、MT-32専用) ❺他の音源と差し替えに便利なCM&MT音源専用ミュート機能 ❻LA音源用音色ライブラリ収録(CM-32L、64、MT-32専用) ❼内蔵FM音源、内蔵ADPCM音源対応 ❽リアルタイム録音機能 ❾ステップ入力、エディット強化(UNDO機能等) ❿国本佳宏氏のDEMO曲とKUNTA氏のグラフィックデータ収録

●本ソフト動作には、メインメモリ2MB及びMIDIボードが必要です。●Mu-1を既にお持ちのお客様には有償バージョンアップサービスを実施します。

新デザイン“MTRウィンドウ”

52chドラムミキサー

LA音源音色エディタ

スピーディーなステップ入力

## Mu-1 Magazine

ミューワンマガジン
Vol.1
8月下旬発売予定 ¥9,800(税抜)

1. ソングファイルクラシック
   特集 モーツァルト(CM-64、MT-32+LA音源対応)
   フィガロの結婚 序曲/魔笛 序曲/ドン・ジョバンニ序曲/ホルン協奏曲 第4番(1楽章 アレグロ・モデラート 2楽章 ロマンツェ・アンダンテ 3楽章 ロンド・アレグロ・ヴィバーチェ)
2. 鳥山敬治LA音源音色ライブラリVol.1
3. KUNTAさんのイラストデータ
4. スタンダードMIDIファイル読み込みコンバート
5. Mu-1 Super体験版
   "Mu-1 聴くだけ"搭載●Mu-1のSNGファイルの再生、チェインプレーが可能●LA音源の音色変更可能
6. その他おまけ機能

*本ソフト動作には、MIDIボードが必要です

HARMONY IN YOUR LIFE
SAN MUSICAL SERVICE
〒154 東京都世田谷区池尻3-21-28 池尻成和ビル 03(3419)8839

●Musicstudio、PRO-68K、Ver2.0はシャープ(株)の登録商標です。
●ミュージ郎はローランド(株)の登録商標です。

New

TAKERU

brother

パソコンソフト自販機

| 地域 | 店名 | 電話 |
|---|---|---|
| 札幌 | そうご電器YES 5F | (011)214-2850 |
| | パソコンショップハドソン | (011)205-1590 |
| 青森 | 電巧堂チェーン青森本店 | (0177)23-2356 |
| 盛岡 | デンコードー盛岡本店 | (0196)54-2772 |
| 秋田 | 電巧堂チェーン秋田駅前本店 | (0188)34-3151 |
| 仙台 | デンコードー仙台本店 | (022)225-5290 |
| | デンコードーDac仙台東口店 | (022)291-4744 |
| | 庄子デンキコンピュータ中央店 | (022)224-5591 |
| 郡山 | うすい百貨店 | (0249)32-0001 |
| 福島 | 庄子デンキ粉又駅前店 | (0245)21-2011 |
| いわき | いわきマイコンショップ | (0246)23-0513 |
| | Dac平店 | (0246)21-8585 |
| 山形 | 庄子デンキ山形七日町本店 | (0236)42-1222 |
| 新潟 | PIC | (025)243-5135 |
| | PICこばり店 | (025)233-5791 |
| 長岡 | 丸専デパート 1F | (0258)33-4970 |
| | 丸専マイコンショップジャスコ長岡店 | (0258)27-6033 |
| 宇都宮 | K&P宇都宮 | (0286)62-0002 |
| 前橋 | パソコンランド21 前橋店 | (0272)21-2721 |
| 高崎 | パソコンランド21 高崎店 | (0273)26-5221 |
| 太田 | パソコンランド21 太田店 | (0276)45-0721 |
| 桐生 | パソコンランド21 桐生店 | (0277)45-2721 |
| 水戸 | 川又書店駅前店 2F | (0292)31-0102 |
| つくば | ミドリ電気館 | (0298)55-3715 |
| 大宮 | ダイエー大宮店 6F | (0486)45-4147 |
| 川越 | サン家電川越本店 | (0492)44-5461 |
| 所沢 | サン家電所沢店 2F | (0429)39-5117 |
| 松戸 | イトーヨーカドー松戸店 | (0473)68-5131 |
| 千葉 | ラオックス千葉店(PERIE 4F) | (0472)27-5318 |
| 八千代台 | ラオックス八千代台店 | (0474)85-2261 |
| 秋葉原 | 丸善無線ECCS 4F | (03)3255-4911 |
| | ミナミ電気館 4F | (03)3255-4040 |
| | サトームセン(メディアセンター) | (03)5256-3267 |
| | サトームセン(ラジオ館) 5F | (03)3251-1464 |
| | サトームセン本店 5F | (03)3253-5879 |
| | CVA秋葉原 2F | (03)3258-3711 |
| | コム本店 | (03)3251-1523 |
| | ザ・コンピュータ館 | (03)5256-3111 |
| 新宿 | ラオックス新宿店 | (03)3350-1241 |
| 池袋 | ビックカメラ池袋東口本店 4F | (03)3988-0020 |
| | ビックカメラ池袋北口店 4F | (03)3988-8666 |
| 渋谷 | 上新電機J&P渋谷店 1F | (03)3496-4141 |
| | ビックカメラ渋谷店A館 6F | (03)3477-0002 |
| 武蔵野 | ラオックス吉祥寺店 2F | (0422)21-3471 |
| 小金井 | サン家電小金井店 | (0423)85-3810 |
| 立川 | 丸善無線立川店(Will 6F) | (0425)27-6211 |
| 八王子 | ムラウチ電気 2F | (0426)42-6211 |
| | J&P八王子店(そごう 7F) | (0426)26-4141 |
| 町田 | 東急ハンズ町田店 B1F | (0427)28-2603 |
| | J&P町田店 | (0427)23-1313 |
| 横浜 | ソフトクリエイト横浜店 | (045)314-4777 |
| | 横浜VIVRE 21 7F | (045)314-2121 |
| | ダイエー戸塚店 3F | (045)881-1261 |
| | ICコスモランドあざみの | (045)901-1901 |
| | ICコスモランドかもい | (045)934-9636 |
| | ビックカメラ横浜店 | (045)320-0002 |
| 川崎 | セキグチデンキ館 | (044)244-5421 |
| 藤沢 | WAVE EYE | (0466)43-1771 |
| 平塚 | ㈱梅屋 | (0463)22-4147 |
| 厚木 | ラオックス厚木店オーディオ館 | (0462)22-2722 |
| 大和 | WAVE EYE大和店 | (0462)63-9898 |
| 相模原 | サトームセン相模大野店 | (0427)41-7786 |
| 長野 | ダイエー長野店 7F | (0262)27-1311 |
| 松本 | 遠兵 | (0263)32-6350 |
| 金沢 | うつのみや片町店 | (0762)21-6136 |
| 富山 | 丸の内カラー駅前店 | (0764)41-9075 |
| 福井 | エジソン | (0776)26-2228 |
| | だるまや西武 7F AVC | (0776)27-0111 |
| 沼津 | メルバ沼津 | (0559)22-4858 |
| 静岡 | メルバ静岡 | (0542)54-5338 |
| 富士 | メルバ富士店 | (0545)51-8022 |
| 浜松 | メルバ浜松本店 | (0534)64-8412 |
| | ホーエー家電有楽店 | (0534)53-1441 |
| | マエダM&V | (0534)73-1691 |
| 名古屋 | 栄電社テクノ名古屋 | (052)581-1241 ★ |
| | パソコンショップコムロード | (052)263-5828 ★ |
| | ちくさ正文館書店ターミナル店 | (052)732-3601 ★ |
| | カトー無線電機電気館 4F | (052)264-1534 ★ |
| | 丸善無線 第一アメ横店 | (052)263-1626 ★ |
| | 栄電社テクノ大須 | (052)252-0181 ★ |
| | トップカメラ 4F | (052)971-0111 ★ |
| | J&P大須店 | (052)262-1141 ★ |
| | エイデンメディア大須 | (052)242-8591 ★ |
| 豊橋 | 栄電社テクノ豊橋 | (0532)52-1231 |
| 岡崎 | ジャスコ岡崎店 3F | (0564)23-4960 |
| 豊田 | ジャスコ豊田店パソコンショップJPC | (0565)35-1761 |
| 岐阜 | パソコンショップコムロード岐阜店 | (0582)66-0288 |
| 大垣 | スイテック大垣ヤナゲン店A館 6F | (0584)81-3491 |
| 津 | Kawai無線OA 津店 | (0592)26-0111 |
| 四日市 | Kawai無線カデンワールド四日市 | (0593)54-3366 |
| 松坂 | Kawai Bax店 6F | (0598)26-0111 |
| 大津 | 西武百貨店関西大津 | (0775)25-0111 |
| 日本橋 | J&Pテクノランド | (06) 634-1211 |
| | J&Pメディアランド | (06) 634-1511 |
| | ニノミヤエレランド | (06) 632-2038 |
| | ニノックスコア日本橋店 | (06) 647-2038 |
| 難波 | ニノミヤパソコンランド難波店 | (06) 643-3217 |
| | J&Pコスモランド | (06) 634-3111 |
| 梅田 | ニノミヤパソコンランド駅前第4ビル店 | (06) 341-2031 |
| | J&P阪急三番街店 | (06) 374-3311 |
| 高槻 | 上新電機J&P高槻店 | (0726)85-1212 |
| 八尾 | 西武百貨店関西八尾店 | (0729)97-0111 |
| 京都 | J&P京都近鉄店 | (075)341-5769 |
| 和歌山 | J&P和歌山店 | (0734)28-1441 |
| 神戸 | 星電社三宮本店C-ソフト | (078)391-8171 |
| | ダイエー三宮電器館バレックス | (078)391-7911 |
| | 上新電機三宮一番館 | (078)211-2111 |
| 姫路 | 上新電機J&P姫路店 | (0792)22-1221 |
| | 姫路コンピュータソフトコンパックス | (0792)94-8244 |
| 岡山 | 岡山VIVRE 21 | (0862)32-8881 |
| 倉敷 | 倉敷ファソランド | (0864)25-4701 |
| 広島 | 松本無線パーツ | (082)243-4451 |
| | ダイイチパソコンCITY | (082)248-4343 |
| 高松 | Sound Check MOVE | (0878)61-6171 |
| 徳島 | 徳島そごう 7F | (0886)25-4056 |
| 高知 | デンキのタグチ | (0888)23-0181 |
| 松山 | ダイイチ松山パソコンCITY | (0899)31-6711 |
| 北九州 | ベストマイコン小倉パソコン館 | (093)551-6281 |
| 福岡 | ベストマイコン福岡店 | (092)781-7131 |
| | 寿屋エレデ博多 | (092)281-4411 |
| 長崎 | ベスト電器長崎新地店 | (0958)28-1333 |
| 熊本 | 寿屋本荘店 | (096)372-5411 |
| | ベストマイコン熊本パソコン館 | (096)332-4180 |
| 鹿児島 | ベスト電器鹿児島パソコン館 | (0992)23-2081 |
| 大分 | ベスト電器大分パソコン館 | (0975)32-9396 |
| 宮崎 | 宮崎寿屋百貨店 7F | (0985)27-4111 |
| 那覇 | ベスト電器那覇店 6F | (0988)62-7588 |

★印のお店は、NEW TAKERUに変わりました。
機能アップして、みんなを待ってるよ。会いに来てね！

# アートディンクフェア

## アートディンクのシミュレーションゲームが TAKERUで続々登場！

SIMULATION GAME
RAILROAD MANAGEMENT
URBAN DEVELOPMENT

A.III.

A列車で行こう

好評発売中

レールを敷き、駅をつくる。19種類の列車を走らせる。
もちろん、効率的なダイヤを組んでだ。土地を買い、資材を運び、工場を建てる。ビル、デパート、スタジアム……。
四季の移り変わりとともに、街はやがて都市となる。
「AⅢ」がついにX68000の世界へやってきた……！

X68000……**¥9,800**(税込)

「AⅢマップコンストラクション」 **¥3,000**(税込)
「AⅢマップコンストラクション・新マップ付」 **¥4,000**(税込)

## X68000シリーズ 好評発売中！

FAR SIDE MOON 地球防衛軍Ⅱ the earth self defense force **¥7,000**(税込)

機甲師団 **¥7,000**(税込)

栄冠は君に 高校野球全国大会 **¥7,000**(税込)

●A列車で行こうⅡ新マップ付……**¥8,000**
A列車で行こうⅡ新マップのみ……**¥2,500**
●大海令……**¥8,000**
大海令シナリオDE……**¥2,500**
大海令シナリオFG……**¥2,500**
●ダブルイーグル……**¥7,000**
ダブルイーグルトリッキーホール……**¥2,000**
●南海の死闘……**¥6,000**
南海の死闘シナリオ……**¥2,500**

※価格はすべて消費税込みの価格です。

## 「TAKERU CLUB」会員大募集！

「TAKERU CLUB」はTAKERUのユーザーの皆様で作るクラブです。
メンバーの方には、その証であるTAKERU CLUB CARDが贈られるとともに、
NEW TAKERUをご利用して頂くことで様々な特典が受けられます。

●特典
①カードを挿入するだけで自分の持っている機種の最新情報が見れる！
②購入したソフトの詳細マニュアルが自動的にしかも早く届く！
③ユーザー登録も簡単！
④TAKERU クーポンシステムでソフトがもらえる！
その他「CLUB会報」、イベント情報が届きます。
お問い合わせは、TAKERU事務局まで。

NEW TAKERU 新登場記念特別入会キャンペーン実施中！ 8.31まで！

入会申込書はお近くのTAKERUにおいてあります。

期間中の入会なら、年会費(500円)と入会費(500円)が無料の上、タケルクーポン券10点(1,000円相当)の特典つきだ！


ブラザー工業株式会社 〒467 名古屋市瑞穂区苗代町2番1号 新事業推進室
TAKERU事務局 (052)824-2493
東京営業所(03)3274-6916
大阪営業所(06)252-4234


通信販売 通信販売をご希望の方は、ソフト名・機種名・住所・氏名・電話番号を明記の上TAKERU事務局まで現金書留でお申し込み下さい
代金引換は一度、現金書留で申し込んで頂いた方に案内させて頂きます

KONAMI

# それでも君は、

# 生中継68™ に

「生中継68」の前評判が高いらしいが、オレはだまされないぞ。
というのは、以下の９つの事実を知っているからだ。

①「生中継68」のゲーム画面は攻撃側と守備側に分割される。
画期的な画面構成だというが、これでは他の理球ゲームで養ったカン
が通用しないのではないか。不安だ。
②「生中継68」のエディットモードは、設定する項目が細かすぎる。
エディットしなくてもちゃんとゲームは楽しめるのだから、ここまで
緻密な設定機能がはたして必要なのかどうか。
③「生中継68」では、チームのユニフォームの色やデザインを選べるの
は当然としても、それにあわせてチームマークのカラーまでコーディ
ネイトされてしまうのは大きなお世話ではないか。
④「生中継68」のオープニングデモは確かに迫力はある。が、ゲーム自
体がきちんと面白いのだから、こんなところまで凝るのはヤリ過ぎで
はないか。
⑤「生中継68」のグラフィックはリアルすぎる。
例えばデッドボールはどうするのだ。こんなリアルな映像で再現され
るのかと思うと、おお、観る前から痛さに寒気がするではないか。
⑥「生中継68」はキーボードを使わなくても、全てのエディット操作を
マウスでできるように設定されている。
マウスを使っていない人の胸の痛みを考えたことがあるのか。
⑦「生中継68」には「操作方法ミニＰＯＰ」がついているそうだ。
そんなオマケをサービスするというのは、企業努力を見せびらかした
いシタゴコロが見え見え。かた腹痛いわ。
⑧「生中継68」のゲーム性は「68版野球ゲームの決定版」を狙った完成
度だそうだが、それは商売上の理由よりも開発担当者の個人的なこだ
わりでやっているとしか思えない。
⑨この広告だってヘンだ。

※操作方法ミニＰＯＰ…パッケージの中に入っているオマケ。
謙遜ではなく、本当に大したモノではな
この場合のＰＯＰは、Point Of Play の

コナミ株式会社 東京本社／〒101 東京都千代田区神田神保町3丁目25 大阪支店／〒561 大阪府豊中市庄内栄町4丁目23-18
札幌営業所／〒060 札幌市中央区北一条西5丁目2-9 福岡営業所／〒810 福岡市中央区天神2丁目8-30

# 期待するのか?

それでも期待してくれる人は、開発者にはげましのお便りを出そう!

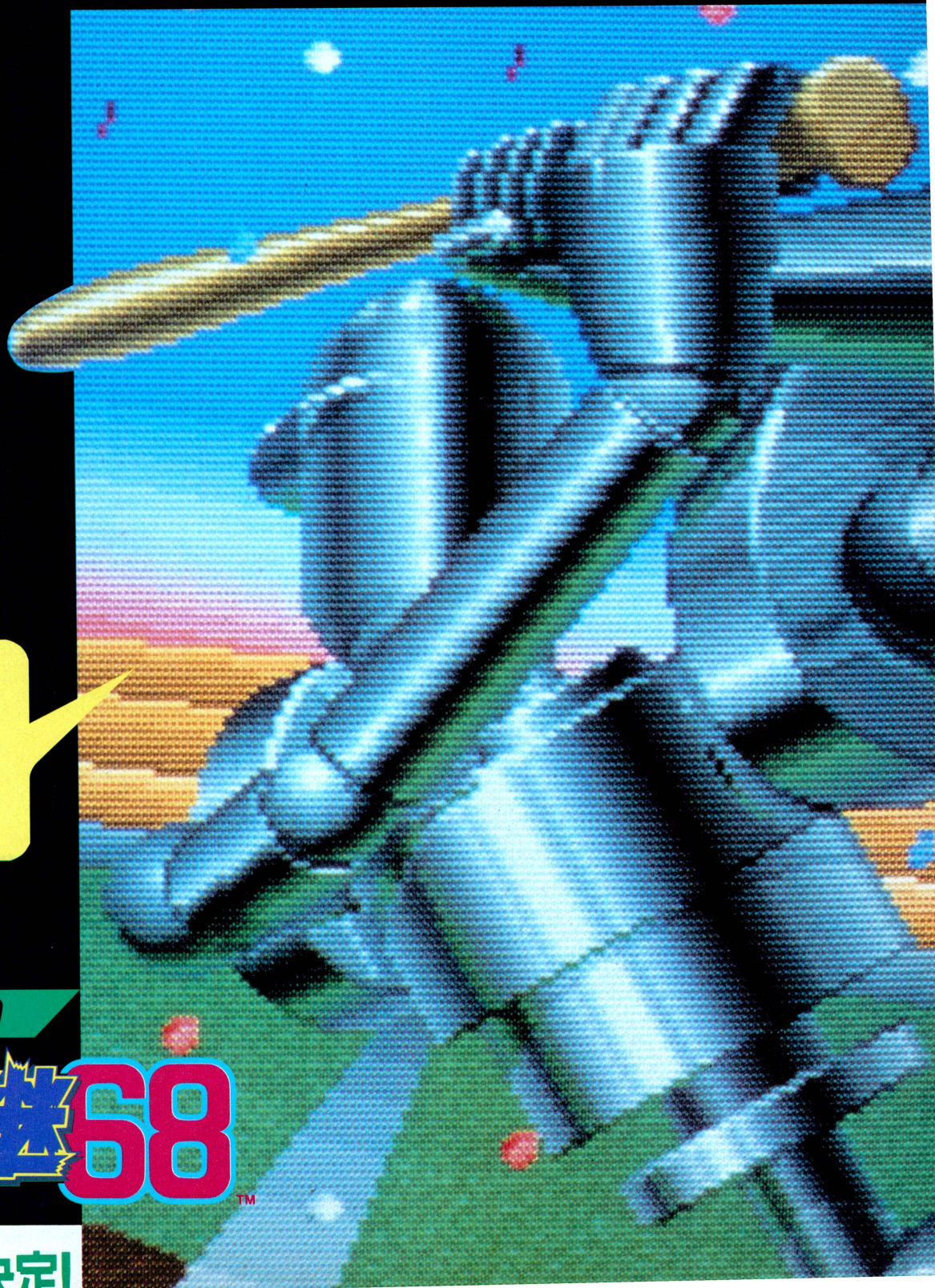
X68000

生中継68™

7月19日発売決定!

# シロウトには、危険です。

あの〈ドラッケン〉の高精度紋様のために、[illegible]りがまで駆使した決定版。このハイパー・リアルな3-D RPGは、ゲームのプロに試していただきたい。

X68000版 近日発売

DRAKKHEN™

全米ヒットチャートNo.1
〈ドラッケン〉
税抜価格9,700円

〈スペックは挑発する〉
256色グラフィック表示によるハイパー・リアル3-D
オリジナルにもなかったモンスターの声を収録
オリジナルを超える3-D超高速360°スクロール
FM+PCMによるハイパー・リアル・サウンド
日・英・独・仏・伊 5ヵ国語表示
24時間リアルタイム進行
4戦士同時アクション機構

epic
INFOGRAMES

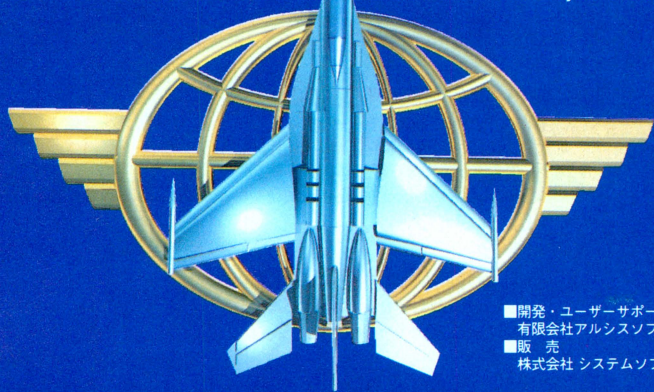

発売元：EPIC/SONY RECORDS 〒107 東京都港区南青山1-1-1 新青山ビル西館8階 (TEL)03-3475-2632

Epic/Sony Recor
A Division of Sony Music Entertainment (Japan

# システムソフトが広げる、面白くする、
# X68000エキサイティング・シーン。

SystemSoft X68000series Line up

## 大戦略III'90 NINETY

This war simulation program is not only more detailed but faster and easier to use.
The latest and greatest simulation quality in our great strategy series.
Makes you feel like you're right there in the heat of battle commanding weapons
and troops using your exceptional personal skills.
You will use the most advanced strategies to overcome and defeat your enemies.

■開発・ユーザーサポート
有限会社アルシスソフトウェア
■販　売
株式会社 システムソフト

### 最強の戦士へ贈ろう、
### 栄光のエンブレム。

戦略シーン、未知なる次元へ。シミュレーションゲーム史上不朽の名作〈大戦略シリーズ〉の最高峰「大戦略III'90」が、遂に待望のX68000に登場。コンピュータ側がより詳しい戦況を把握する戦略思考ルーチンの導入、ゲームの同時進行を可能にするリアルタイムオペレーションをはじめ、可変ウィンドウの採用、メニューやコマンドのシンプル化などによる画期的なシステムを実現。ビジュアルもより美しく進化し、ゲームを盛り上げる迫力のBGMも加わった。しかも、98シリーズのマップおよびゲームデータも活用可能。これまでの開発ノウハウを結集した、まさにシリーズの頂点。最強の敵を迎え、いま栄光のエンブレムを賭けた熾烈な戦いが始まる。

開発中

※画面は開発中のものです。

■X68000 シリーズ　■5"-2HD（3 枚組）
●アナログRGB（31KHz 対応）ディスプレイをお使いください。
●入力装置として、X68000 添付のマウスを使用します。

※「大戦略III'90」X68000 版に限りまして、技術的な内容などユーザーサポートに関するお問い合わせはアルシソフトウェア、販売に関するお問い合わせはシステムソフト営業部までお願いします。

(有)アルシスソフトウェア：佐世保市松浦町5-13　グリーンビル３Ｆ
〒857　TEL 0956-22-3881

**価格 9,800 円**

---

## BRETONNE LAIS

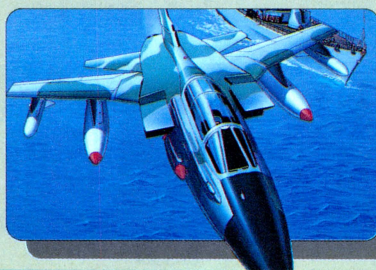
ブルトン・レイ シナリオエディタ

### 剣と、魔法と、ロマンスと。
### 壮大な冒険物語を、君の手で創造。

■X68000 シリーズ　■5"-2HD（2 枚組）
●別売の「ブルトン・レイ」X68000版が必要です。
●アナログRGB（31KHz 対応）ディスプレイをお使いください。
●メインメモリが2MB 以上の場合、日本語入力フロントプロセッサとしてASK68Kが使用できます。（2MB 未満の場合は、単漢字入力となります。）

**価格 5,800 円**

---

## 大戦略II キャンペーン版 CAMPAIGN VERSION

### 果てしなき戦いが、いま始まる。

8ステージ連続制覇に挑む
〈キャンペーンモード〉導入。
君は、どこまで戦えるか。

■X68000 シリーズ　■5"-2HD（2 枚組）
●アナログRGB（31KHz 対応）ディスプレイをお使いください。

**価格 9,800 円**

---

## BOMBER MAN
ボンバーマン

© Original Work 1990 HUDSON SOFT
© Derivative Work 1990 SystemSoft

### コーフンのバクハツだ！
### 愉快な爆弾アクションゲーム。

■X68000 シリーズ　■5"-2HD
●アナログRGB（31KHz 対応）ディスプレイをお使いください。
●アタリ社仕様の2トリガージョイスティック、ジョイパッドが使用できます。
●3人以上でプレイする場合は上記のジョイスティック、ジョイパッドが必要です。

**価格 7,800 円**

---

発売日等の最新情報を下記のとおりテレフォンサービスにてご案内いたしております。どうぞお気軽にご利用ください。

新製品の発売日および内容のご案内は…
テレフォンサービス専用電話　東京：03-3326-8710
福岡：092-752-2602

商品のお申し込みおよび発売日に関するお問い合わせは…
営業部専用電話　092-752-5262
土曜日、日曜日、祝祭日は営業いたしておりません。

商品に関する技術的なお問い合わせは…
ユーザーサポート専用電話　092-752-5278
月～金　9:00～12:00　13:00～17:00（祝祭日を除く）

●総合カタログをご希望の方は請求券をはがきに貼り、住所・氏名・年齢・電話番号・使用機種名を明記の上、弊社宛にご送付ください。
●製品の仕様は、機能・性能の改善のため将来予告なしに変更することがあります。
●表示価格に消費税は含まれておりません。

SystemSoft 株式会社 システムソフト
アミューズメント事業部
〒810 福岡市中央区天神3丁目10-30

総合カタログvol.10請求
Oh!X　7月号
有効期限：1991年7月末

# 全国通販

SHARP 認定 PPO-SHOP **O.A.ランド**

(TEL) 03-3770-8855

■アフターサービス万全のサポート体制

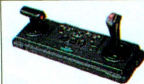
●下取・買取は電話で見積りしております。責任を持って下取りさせて頂きます。

営業時間
平日………AM10:00〜PM7:00
土日・祭日…AM10:00〜PM6:00

▶6・15 〜7・14

SHARPのことならなんでおまかせ!!

大徳買セール!安く値切ってネ。(本体セット:送料 消費税込み)
お電話下さい。㊙価格をお知らせいたします。

流通事情により、広告表示価格は、お安くなる場合がありますので、ドンドンお電話下さい。

CYBER STICK
■CZ-8NJ2
(定価¥23,800)
OAランド特価 ▶¥18,000

電子手帳 ●見やすい漢字4桁表示!! 情報任時代の必需品!!
■PA-9500(¥48,000)…▶特価¥38,000
■PA-8500(¥28,000)…▶特価¥15,000
■PA-7500(¥22,000)…▶特価¥12,000

低金利クレジットをご利用下さい。平日AM10時〜PM7時、土日・祭日AM10時〜PM6時迄ガンバッテま〜す!!

## SHARP X68000シリーズセット(送料・消費税込み)

### X68000XVI NEW

①CZ-634C-TN+CZ-613D-TN
定価合計¥503,000

| 12回 | ¥33,600 | 24回 | ¥17,800 |
|---|---|---|---|
| 36回 | ¥12,400 | 48回 | ¥ 9,700 |

②CZ-634C-TN+CZ-606D-TN
定価合計¥447,800

| 12回 | ¥30,000 | 24回 | ¥15,900 |
|---|---|---|---|
| 36回 | ¥11,100 | 48回 | ¥ 8,700 |

### X68000XVI-HD

①CZ-644C-TN+CZ-613D-TN
定価合計¥653,000

| 12回 | ¥44,200 | 24回 | ¥23,500 |
|---|---|---|---|
| 36回 | ¥16,300 | 48回 | ¥12,800 |

②CZ-644C-TN+CZ-606D-TN
定価合計¥597,800

| 12回 | ¥39,900 | 24回 | ¥21,200 |
|---|---|---|---|
| 36回 | ¥14,700 | 48回 | ¥11,600 |

■CZ-634C
特価¥TEL下さい!!
■CZ-644C
特価¥TEL下さい!!

XVIセットお買上げの方に①ニュージーランドストーリー ②V-BALL ③ジョイカード(連射式)④ディスケット20枚プレゼントいたします。!!

### X68000SUPER

①CZ-604C-TN+CZ-613D-TN
定価合計¥483,000

| 12回 | TEL下さい | 24回 | TEL下さい |
|---|---|---|---|
| 36回 | TEL下さい | 48回 | TEL下さい |

②CZ-604C-TN+CZ-606D-TN
定価合計¥427,800

| 12回 | TEL下さい | 24回 | TEL下さい |
|---|---|---|---|
| 36回 | TEL下さい | 48回 | TEL下さい |

### X68000SUPER-HD

①CZ-623C-TN+CZ-613D-TN
定価合計¥633,000

| 12回 | TEL下さい | 24回 | TEL下さい |
|---|---|---|---|
| 36回 | TEL下さい | 48回 | TEL下さい |

②CZ-623C-TN+CZ-606D-TN
定価合計¥577,800

| 12回 | TEL下さい | 24回 | TEL下さい |
|---|---|---|---|
| 36回 | TEL下さい | 48回 | TEL下さい |

■CZ-604C
特価¥TEL下さい!!
■CZ-623C
特価¥TEL下さい!!

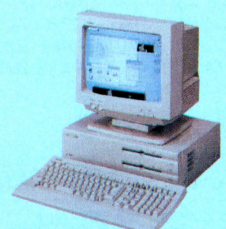
### X68000PROII

①CZ-653C+CZ-613D
定価合計¥420,000

| 12回 | TEL下さい | 24回 | TEL下さい |
|---|---|---|---|
| 36回 | TEL下さい | 48回 | TEL下さい |

②CZ-653C+CZ-605D
定価合計¥400,000

| 12回 | TEL下さい | 24回 | TEL下さい |
|---|---|---|---|
| 36回 | TEL下さい | 48回 | TEL下さい |

③CZ-653C+CZ-606D
定価合計¥364,800

| 12回 | TEL下さい | 24回 | TEL下さい |
|---|---|---|---|
| 36回 | TEL下さい | 48回 | TEL下さい |

### X68000PROII-HD

①CZ-663C+CZ-613D
定価合計¥530,000

| 12回 | TEL下さい | 24回 | TEL下さい |
|---|---|---|---|
| 36回 | TEL下さい | 48回 | TEL下さい |

②CZ-663C+CZ-605D
定価合計¥510,000

| 12回 | TEL下さい | 24回 | TEL下さい |
|---|---|---|---|
| 36回 | TEL下さい | 48回 | TEL下さい |

③CZ-663C+CZ-606D
定価合計¥474,800

| 12回 | TEL下さい | 24回 | TEL下さい |
|---|---|---|---|
| 36回 | TEL下さい | 48回 | TEL下さい |

■CZ-653C
特価¥TEL下さい!!
■CZ-663C
特価¥TEL下さい!!

上記組合せのディスプレイ(モニター)変更自由!!
詳しくは、お電話にてお問い合せ下さい!!

■期間中、セットでお買い上げの方には、①ジョイカード(連射式)と②テトリスやドルアーガの塔などの入ったゲームパックをプレゼント!!

## 周辺機器コーナー 電話で値切ろう。

### プリンターセットコーナー

①CZ-8PC5 NEW 定価¥96,800
●48ドット●熱転写カラー 漢字プリンター
大特価¥69,500

②CZ-8PK10(24ピン漢字プリンター136桁)
定価¥97,800…特価¥71,000

③CZ-8PG1(24ピンカラー漢字プリンター80桁)
定価¥130,000…特価¥93,000

④CZ-8PG2(24ピンカラー漢字プリンター136桁)
定価¥160,000…特価¥114,000

### OAランド特選品!!

■IO-735X(定価¥248,000)
●カラーイメージジェットプリンター
特価¥170,000

### X68000用ハードディスク

■SCSIタイプ
●アイテック
①TX-80S (¥108,000)……特価¥ 81,000
②TX-130S (¥138,000)……特価¥103,000
③TX-180S (¥185,000)……特価¥138,000
●テクノジャパン
①PD-50GS (¥116,000)…特価¥ 81,000
②PD-100GS (¥148,000)…特価¥101,000
③PD-130GSX(¥168,000)…特価¥114,000

■SASIタイプ
●ロジテック
①SHD-40(¥99,800)……特価¥ 60,000

※X68000SUPER/XVI以外の機種では、SCSIボードが必要となります。
★SCSIボード………特価¥ 22,000
★光ディスク…………特価¥320,000

### X68000用周辺機器コーナー

①CZ-6VT1(カラーイメージユニット)
定価¥69,800…特価¥ 51,500
②CZ-8NS1(カラーイメージスキャナー)
定価¥188,000…特価¥135,000
③CZ-6BM1(MIDIボード)
定価¥26,800…特価¥ 20,000
④CZ-6BE2A(2MB増設RAMボード)
定価¥59,800…特価¥ 45,000
⑤CZ-6BE2B(2MB増設RAM)
定価¥54,800…特価¥ 41,500
⑥CZ-6BP2(数値演算プロセッサ)
定価¥45,800…特価¥ 34,800
⑦CZ-6EB1(拡張I/Oボックス=4スロット)
定価¥88,000…特価¥ 65,000
⑧CZ-6BP1(数値演算プロセッサボード)
定価¥79,800…特価¥ 59,000

### 《計測技研》増設メモリ&プロセッサ

●高速増設メモリと数値演算プロセッサが一つのボードになった!!●

● KGB-X68 PRK-00(¥ 34,000)…特価¥26,000
● PRK-01(¥ 58,000)…特価¥43,500
● PRK-02(¥ 74,000)…特価¥55,500
● PRK-03(¥ 98,000)…特価¥73,500
● PRK-04(¥122,000)…特価¥91,500
● PRK-10(¥ 72,000)…特価¥54,000
● KGB-X68 PRK-11(¥ 96,000)…特価¥ 72,000
● PRK-12(¥112,000)…特価¥ 84,000
● PRK-13(¥136,000)…特価¥102,000
■KGB-X68PRK-14(¥160,000)…特価¥115,000
■MC6888 1RC16(¥38,000)……特価¥ 28,500

### I・Oデータ増設RAMボード

■PIO-6BE1-A (1MB) 定価¥25,000 特価¥17,300
■PIO-6BE2-2M (2MB) 定価¥50,000 特価¥33,500
■PIO-6BE4-4M (4MB) 定価¥88,000 特価¥58,500

### ■OAランド推奨ソフト

Ⓐ Easy Paint SX 68K (CZ-263GW) 特価TEL下さい!!
Ⓑ Music studio PRO 68K (CZ-261MS) 定価¥28,800 特価TEL下さい!!
Ⓒ Print Shop V.2 (CZ-265HS) 特価TEL下さい!!
Ⓓ Multiword PRO 68K (CZ-225BS) 定価¥32,000 特価¥24,000
Ⓔ CZ-245LS (C-コンパイラII) 定価¥44,800 特価¥33,500
Ⓕ Telepotion PRO 68K (CZ-258BS) 定価¥22,800 特価¥18,000

## 通信販売のご案内

全国通販

■銀行振込で申し込みの方は商品名及びお客様の住所・氏名・電話番号をお知らせ下さい。
〔振込先〕第一勧業銀行 渋谷支店
普通No.1163457 ㈱オーエーランド

■年中無休です!!

■現金書留で送金されるお客様は電話番号と商品名、数量を明記して同封して下さい。■クレジットでご購入を希望される方は申し込み用紙をお送り致しますのでご記入の上返送して下さい。20才以上の方は、原則として保証人不要です。クレジットは1〜60回払で月々5,000円よりご自由に設定できます。

クレジット表

| 3回 | 3.5% | 6回 | 4.5% | 10回 | 6% | 12回 | 6% | 15回 | 8.5% | 18回 | 11% | 20回 | 12% |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 24回 | 12.5% | 30回 | 17% | 36回 | 17.5% | 42回 | 22.5% | 48回 | 23% | 54回 | 29% | 60回 | 29.5% |

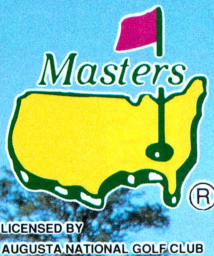

㈱オー・エー・ランド
〒150 東京都渋谷区桜丘町3-13 アルカディア2F
☎(03)3770-8855
FAX(03)3770-7080


関東エリアの送料は、1個につき¥1,000です。

★全商品保証書付。専門のアドバイザーが、お客様のニーズに対応します。
★初期不良・輸送トラブル等に迅速に対応し、即交換させていただきます。

■表示価格は、税別表示です。詳しくは、お電話にて、お問い合せ下さい。掲載の価格は、5月下旬現在です。

Masters

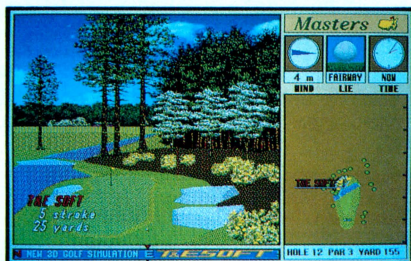
LICENSED BY
AUGUSTA NATIONAL GOLF CLUB

NEW 3D GOLF SIMULATION

# 遥かなるオーガスタ™

はるかなるオーガスタ

## オーガスタ・ナショナル・ゴルフ・クラブと正式契約

**X68000版 好評発売中!!**

(5"2HD 3枚組)
要2M RAM
■標準価格 **¥12,800**(税別)

- ●実際にゴルフコースに立った状態と同じ視野でプレイ可能
- ●どの地点にきても全方向の視野画面をリアルタイム3D表示
- ●ホールすべてにアンジュレーション(起伏)を3Dで表示
- ●ボールの落下地点の状態によってバウンド、転がり等が本物同様に変化
- ●ストロボアクションモードでボールの軌跡を確認可能
- ●バックスピン・トップスピンも可能
- ●プレイモードは3種類。ストロークプレイ、マッチプレイ、トーナメントプレイ
- ●キャディーは4人の中から選択
- ●初心者でも気軽に楽しめるスローモード機能あり
- ●スコア・各種個人データ等を自動保存、プリントアウトも可能
- ●マウスのみですべて簡単操作
- ●31KHz/15KHz両モード対応、ビデオ出力で大画面プレイ可能
- ●ADPCMによるリアルなサウンド
- ●その他機能満載

## コースデータVol.2

NEW 3D GOLF SIMULATION

# EIGHT LAKES G.C.™

EIGHT LAKES G.C.

T&E SOFT ORIGINAL COURSE

**エイトレイクスG.C.は、その設計に充分な時間をかけ、細部にわたって練り上げられた戦略重視のオリジナルコースです。**

- ●8つの湖が効果的にレイアウトされた美しい18ホール
- ●湖、森林、谷、丘陵、そして砂浜等、変化に富んだ難コース
- ●目にも鮮やかな桜並木が水面に影を落す
- ●各ホールとも多彩な攻め方が要求される、戦略重視のテクニカルコース
- ●速いグリーンがより高度なパッティングテクニックを要求する
- ●国際色豊かな4人の女性キャディーが登場

"スリーアイランズ"と呼ばれる名物ショートホール

**X68000版 7月5日(金)発売**

(5"2HD 2枚組) ■標準価格 **¥5,800**(税別)

※「EIGHT LAKES G.C.」は、「遥かなるオーガスタ」が必要です。

POLYSYS™ Integrated 3D Processor

このマークはT&E SOFTの商標です
POLYSYS搭載の3Dソフトには、このマークが表示されます

Technology & Entertainment Software
T&E SOFT
株式会社 ティーアンドイーソフト
〒465 名古屋市名東区豊が丘1810番地 PHONE:052-773-7770

No Copy
このマークは不法コピー禁止マークです
ソフトウエア法的保護監視機構

●3Dゴルフに関するお問い合わせは、NEW 3D GOLF事務局まで PHONE:052-773-7757

超低金利クレジットOK

# 注目!!

夏のボーナス一括払い
手数料(金利)無料
(平成3年7月末をご利用下さい)

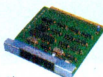
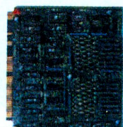
またまた **秋葉原**で おなじみの

# 6/15~7/15

●お近くの方はお
●本体単品で特
●ビジネスソフト定

X68000用ハードディスク
■ITX-680(アイテック)
定価¥198,000
超特価 ¥78,000
(送料・消費税込み¥81,370)

Fine Scanner-X68
(HAL研究所)X68000専用
■HGS-68(定価¥39,800)
特価¥26,000
(送料・消費税込み¥27,295)

X68000シリーズ専用 特価¥14,300
MIDIインターフェースボード
SX-68M(サコム)
(純生コンパチ)定価¥19,800
(送料・消費税込み¥15,244)

X68000メモリボード(シャープ&I/O・DATA)(送料¥500)
①CZ-6 BE1(600C用)定価¥35,000
(送料・消費税込¥27,295)…特価¥26,000
②PIO-6BE1-A 定価¥25,000
(送料・消費税込¥17,819)…特価¥16,800
③PIO-6BE2-2M 定価¥50,000
(送料・消費税込¥34,505)…特価¥33,000
④PIO-6BE4-4M 定価¥88,000
(送料・消費税込¥58,710)…特価¥56,500

ジョイスティック 送料¥500
●X-1PRO
定価¥9,500▶特価¥7,800
●ASCII STICK
定価¥6,800▶特価¥5,500

P&A超低金利クレジットをご利用ください!!

## X68000-XVI 新発売!! (送料・消費税込み)

NEW

★先着100名様へ。
ゲームソフト(V-BALL ¥7,900)を
プレゼント!!

### X68000-XVI ▶セットでお買い上げの方に●ディスケット10枚●ジョイカード2ケプレゼント中!!

Ⓐセット:CZ-634C-TN+CZ-606D-TN…定価¥447,800▶特価価格はTEL下さい。

| 12回 | ? | 24回 | ? | 36回 | ? | 48回 | ? | 60回 | ? |
|---|---|---|---|---|---|---|---|---|---|

Ⓑセット:CZ-634C-TN+CZ-613D-TN…定価¥503,000▶特価価格はTEL下さい。

| 12回 | ? | 24回 | ? | 36回 | ? | 48回 | ? | 60回 | ? |
|---|---|---|---|---|---|---|---|---|---|

### X68000-XVI-HD ▶セットでお買い上げの方に●ディスケット10枚●ジョイカード2ケプレゼント中!!

Ⓐセット:CZ-644C-TN+CZ-606D-TN…定価¥597,800▶特価価格はTEL下さい。

| 12回 | ? | 24回 | ? | 36回 | ? | 48回 | ? | 60回 | ? |
|---|---|---|---|---|---|---|---|---|---|

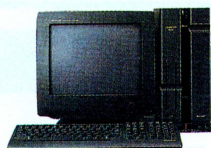
Ⓑセット:CZ-644C-TN+CZ-613D-TN…定価¥653,000▶特価価格はTEL下さい。

| 12回 | ? | 24回 | ? | 36回 | ? | 48回 | ? | 60回 | ? |
|---|---|---|---|---|---|---|---|---|---|

※上記のモニターを、CZ-604D(定価¥94,800)、CZ-605D(定価¥115,000)、CU-21HD(定価¥148,000)に変更の場合、TEL下さい。超特価で販売致します。

## X68000シリーズ ~P&Aスペシャルセット= 台数限定 送料、消費税込み

※セットでお買い上げの方に、●ディスケット10枚、●ジョイカード2個プレゼント中!! 先着100名様。ゲームソフト(V-BALL¥7,900)をプレゼント!!

### SUPER

Ⓐセット:P&A特選セット
■CZ-604C
(本体定価¥348,000)
+
■CZ-606D
(モニター定価¥79,800)
▶P&A超特価 ¥299,000

Ⓑセット
■CZ-604C+CZ-604D
定価¥442,800………▶特価¥305,000
Ⓒセット
■CZ-604C+CZ-605D
定価¥463,000………▶特価¥323,000
Ⓓセット
■CZ-604C+CZ-613D
定価¥483,000………▶特価¥338,000
Ⓔセット
■CZ-604C+CU-21HD
定価¥496,000………▶特価¥346,000

### PRO-II

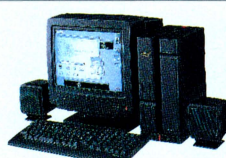
Ⓐセット:P&A特選セット
■CZ-653C
(本体定価¥285,000)
+
■CZ-606D
(モニター定価¥79,800)
▶P&A超特価 ¥242,000

Ⓑセット
■CZ-653C+CZ-604D
定価¥379,800………▶特価¥250,000
Ⓒセット
■CZ-653C+CZ-605D
定価¥400,000………▶特価¥269,000
Ⓓセット
■CZ-653C+CZ-613D
定価¥420,000………▶特価¥283,000
Ⓔセット
■CZ-653C+CU-21HD
定価¥433,000………▶特価¥290,000

### SUPER-HD

Ⓐセット:P&A厳選セット
■CZ-623C
(本体価格¥498,000)
+
■CZ-606D
(モニター定価¥79,800)
▶P&A超特価 ¥382,000

Ⓑセット
■CZ-623C+CZ-604D
定価¥592,800………▶特価¥389,000
Ⓒセット
■CZ-623C+CZ-605D
定価¥613,000………▶特価¥408,000
Ⓓセット
■CZ-623C+CZ-613D
定価¥633,000………▶特価¥420,000
Ⓔセット
■CZ-623C+CU-21HD
定価¥646,000………▶特価¥430,000

### EXPERII

Ⓐセット:P&A厳選セット
■CZ-603C
(本体価格¥338,000)
+
■CZ-606D
(モニター定価¥79,800)
▶P&A超特価 ¥274,000

Ⓑセット
■CZ-603C+CZ-604D
定価¥432,800………▶特価¥280,000
Ⓒセット
■CZ-603C+DZ-605D
定価¥453,000………▶特価¥298,000
Ⓓセット
■CZ-603C+CZ-613D
定価¥473,000………▶特価¥313,000
Ⓔセット
■CZ-603C+CU-21HD
定価¥486,000………▶特価¥321,000

●本広告の掲載の商品の価格については、消費税は含まれておりません。 ●営業時間=平日AM10:00~PM7:00、日祭AM10:00~PM6:00

～84回払いまでOK!!

★頭金なし!★即日発送

●価格は流通事情により変動致しますので、銀行振込・書留等の送付前に、あらかじめお電話にてご確認下さい。

# P&AがズバリP&A超特価セールでご奉仕!!

立寄り下さい。専門係員が説明いたします。
価で受付します。詳しくは電話にてお問合せ下さい。
価の20%引きOK! TELください。

# 全国通販

超特価でクレジットが組める!!

## X68000用ソフトコーナー（送料1ヶ～5ヶまで￥500）

| 商品 | 定価 | 特価 |
|---|---|---|
| ●Z's STAFF PRO68K Ver.2.0（ツァイト） | 定価￥58,000 | 特価￥38,800 |
| ●Z's TRIPHONY デジタルクラフト（ツァイト） | 定価￥39,800 | 特価￥27,800 |
| ●テラッツォ（ハミングバード） | 定価￥19,400 | 特価￥14,200 |
| ●KAMIKAZE（サムシング・グッド） | 定価￥68,000 | 特価￥44,800 |
| ●C & Professional Pack（マイクロウェアジャパン） | 定価￥58,000 | 特価￥41,000 |
| ●Finai Ver3.2（エーエスピー） | 定価￥38,000 | 特価￥29,600 |
| ●C-compiler PRO68K Ver.2 CZ-245L | 定価￥44,800 | 特価￥33,300 |
| ●CARD PRO68K CZ226BS | 定価￥29,800 | 特価￥21,200 |
| ●XBAS to C CHECKER CZ-260LS | 定価￥9,800 | 特価￥7,400 |
| ●OS-9/X68000 CZ219SS | 定価￥29,800 | 特価￥22,500 |
| ●AI-68K CZ234LS | 定価￥188,000 | 特価￥138,000 |
| ●THE 福袋 V2.0 CZ224LS | 定価￥9,900 | 特価￥7,400 |
| ●SOUND PRO68K CZ-214MS | 定価￥15,800 | 特価￥11,400 |
| ●MUSIC PRO68K CZ213MS | 定価￥18,800 | 特価￥13,400 |
| ●Sampling PRO68K CD215MS | 定価￥17,800 | 特価￥12,700 |
| ●MUSIC-studio PRO68K CZ-252MS | 定価￥15,800 | 特価￥21,400 |
| ●MUSIC-PRO68K〔MIDI〕247MS | 定価￥28,800 | 特価￥20,700 |
| ●New-print Shop 221HS | 定価￥19,800 | 特価￥15,500 |
| ●Communication 223CS | 定価￥19,800 | 特価￥14,200 |
| ●Communication Ver.2 CZ-257CS | 定価￥19,800 | 特価￥15,500 |
| ●C-TRACE68 Ver.3.0（キャスト） | 定価￥98,000 | 特価￥69,000 |
| ●サイクロンEXPRESSα68 | 定価￥98,000 | 特価￥69,800 |
| ●G68K Ver2 PRO | 定価￥22,000 | 特価￥17,500 |
| ●SX-WINDOW CZ-259SS | 定価￥6,800 | 特価￥4,900 |
| ●Gツール（ザインソフト） | 定価￥28,000 | 特価￥18,900 |
| ●たーみのる2（SPS） | 定価￥17,800 | 特価￥13,300 |
| ●マジックパレット（ミュージカルプラン） | 定価￥19,800 | 特価￥14,500 |
| ●Hyper word CZ-251BS | 定価￥39,800 | 特価￥29,600 |

●ゲームソフト20%OFF OK!!（一部ソフト除く）

## X68000用ハードディスク（送料￥1,000）

アイテック（SCSIタイプ）

- ■TX-80 ……… 定価￥108,000▶特価￥80,500
- ■TX-130 ……… 定価￥138,000▶特価￥103,000
- ■TX-180 ……… 定価￥185,000▶特価￥137,000

## プリンター（ケーブル・用紙付）（送料￥1,000）

- ■CZ-8PC5-BK NEW …… 定価￥96,800▶特価￥70,000
- ■CZ-8PK10 …… 定価￥97,800▶特価￥71,000
- ■CZ-8PG2 …… 定価￥160,000▶特価価格はTEL!!
- ■CZ-8PG1 …… 定価￥130,000▶特価価格はTEL!!

## モデムコーナー（送料￥1,000）

■COMSTARZ CLUB24/5
（NEC）定価￥39,800
特価￥26,500（送料・消費税込み ￥28,325）

■MD-24FB5V
（オムロン）定価￥39,800
特価￥27,400（送料・消費税込み ￥29,252）

## 周辺機器コーナー（送料￥500）

| 商品 | 定価 | 特価 |
|---|---|---|
| ①CZ-8NSI | 定価￥188,000 | 特価￥145,000 |
| ②CZ-6VTI | 定価￥69,800 | 特価￥52,500 |
| ③CZ-6TU | 定価￥33,100 | 特価￥24,500 |
| ④BF-68PRO | 定価￥19,800 | 特価￥15,300 |
| ⑤CZ-6BEI | 定価￥35,000 | 特価￥26,000 |
| ⑥CZ-6BEIA | 定価￥38,000 | 特価￥28,600 |
| ⑦CZ-6BE2A | 定価￥59,800 | 特価￥44,200 |
| ⑧CZ-6BE2B | 定価￥54,800 | 特価￥40,800 |
| ⑨CZ-6BFI | 定価￥49,800 | 特価￥38,200 |
| ⑩CZ-6BPI | 定価￥79,800 | 特価￥60,000 |
| ⑪CZ-6BMI | 定価￥26,800 | 特価￥20,300 |
| ⑫CZ-6EBI | 定価￥88,000 | 特価￥66,500 |
| ⑬AN-S100 | 定価￥36,600 | 特価￥28,500 |
| ⑭CZ-6SDI | 定価￥44,800 | 特価￥35,000 |
| ⑮CZ-6BN1 | 定価￥29,800 | 特価￥22,600 |
| ⑯CZ-6BV1 | 定価￥21,000 | 特価￥15,900 |
| ⑰CZ-64H | 定価￥120,000 | 特価￥91,500 |
| ⑱CZ-6BG1 | 定価￥59,800 | 特価￥45,000 |
| ⑲CZ-6BU1 | 定価￥39,800 | 特価￥30,300 |
| ⑳CZ-6PVI | 定価￥198,000 | 特価￥153,000 |
| ㉑CZ-6BS1 | 定価￥29,800 | 特価￥22,300 |
| ㉒CZ-8NJ2 | 定価￥23,800 | 特価￥18,500 |
| ㉓CZ-6BL2 | 定価￥298,000 | 特価￥222,000 |
| ㉔JX-100S | 定価￥89,800 | 特価￥38,800 |
| ㉕JX-220 | 定価￥146,000 | 特価￥107,900 |
| ㉖IO-735X | 定価￥248,000 | 特価￥169,000 |

## P&A特選パソコンラック（送料無料）

①3段￥9,000　②4段￥10,800　③5段￥14,800

全機種＝移動自由（キャスター付）・キーボード収納可（5段のみ）＝1230（H）×600（D）×650（W）

## 中古パソコンはP&Aにおまかせ!!

その場で高価現金買取り・高価下取りOK!!

■まずはお電話下さい。
03-3651-1884、FAX：03-3651-0141

■下取り・買取りでお急ぎの方、直接当社に来店、または、宅急便にてお送り下さい。

●下取りの場合…………価格は常に変動していますので査定額をお電話で確認して下さい。（差額は、P&A超低金利クレジットをご利用下さい。）

●買取りの場合…………現品が着き次第、2日以内に買取り金額を連絡し、振込み、又は書留でお送り致します。

●近郊の方は、P&A本店まで、直接お持ち下さい。即金にて、￥1,000,000までお支払い致します。

## 《便利な超低金利クレジットをご利用下さい》

●月々￥1,000円からOK!!　●ボーナス払いOK（夏冬10回までOK）
●支払い回数 1回～84回　●お支払いは、8ヶ月先からでもOK!!

アフターサービス万全
全商品保証付。専門の担当者がお客様の立場で対応します。
初期不良、輸送トラブルetc.
万が一初期不良、輸送トラブルが発生しました際には、即交換させていただきます。

●定休日/毎週水曜日＝第3水曜（祭日の場合は翌日になります）

## 通信販売お申し込みのご案内

〔現金一括でお申し込みの方〕
●商品名およびお客様の住所・氏名・電話番号をご記入の上、代金を当社まで、現金書留でお送りください。（プリンター・フロッピーの場合、本体使用機種名を明記のこと）

〔銀行振込でお申し込みの方〕
●銀行振込ご希望の方は必ずお振込みの前にお電話にてお客様のご住所・お名前・商品名等をお知らせください。
（電信扱いでお振込み下さい。）

〔振込先〕 住友銀行 新小岩支店
普通預金 1451576 ㈱ピー・アンド・エー

〔クレジットでお申し込みの方〕
●電話にてお申し込みください。クレジット申し込み用紙をお送りいたしますので、ご記入の上、当社までお送りください。
●現金特別価格でクレジットが利用できます。残金のみに金利がかかります。
●1回～84回払いまで出来ます。但し、1回のお支払い額は￥1000円以上。

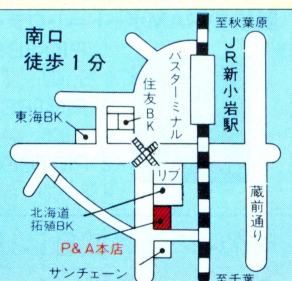
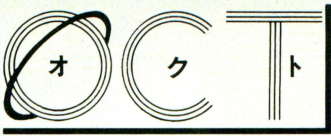
超低金利クレジット率

| 回数 | 3 | 6 | 10 | 12 | 18 | 24 | 36 | 48 | 60 | 72 | 84 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 手数料 | 3.5 | 4.5 | 6.0 | 6.0 | 11.0 | 12.5 | 17.5 | 23.0 | 29.5 | 38.0 | 45.5 |

南口 徒歩1分
至秋葉原
JR新小岩駅
バスターミナル
住友BK
東海BK
リブ
北海道拓殖BK
P&A本店
サンチェーン
蔵前通り
至千葉

●マイコン
●ビデオ
●ビデオテープ

P&A
株式会社ピー・アンド・エー
〒124 東京都葛飾区新小岩2丁目1番地19号
☎03-3651-0148（代）
FAX. 03-3651-0141

営業時間
平日：AM10：00～PM7：00
日祭：AM10：00～PM6：00

●現金書留及び銀行振込でお申し込みの方は、上記商品の料金に3%加算の上でお申し込み下さい。詳しくは、お電話でお問い合せ下さい。

■特に人気のある商品によっては、しばらくお待ち願うことがありますのでご了承下さい!!

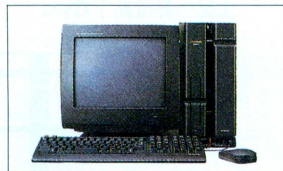
# X68000 ラストチャンス!!

## ■SUPER/PROII/SUPER-HD

◀大人気!!*大戦略II シュミレーションゲーム プレゼント

★JOY CARD (連射式)×2個
★MD-2HD 10枚

(定価¥9,800)

■ビッグバーゲンセール実施中!!ゲームソフト(ビジネス)新製品続々入荷中!!

■SUPER(定価¥348,000)
CZ-604C-TN

■PRO II(定価¥285,000)
CZ-653C-BK/GY

■SUPER-HD(定価¥498,000)
CZ-623C-TN

15型カラーディスプレイTV
CZ-613D-GY/BK
定価¥135,000

14型カラーディスプレー
CZ-606D(GY/BK/TN)
定価¥79,800

21型カラーディスプレイ
CU-21HD
定価¥148,000

CZ-8NJ2 限定
●インテリジェントコントローラ
定価¥23,800
超特価¥18,000

Ⓐ CZ-604C+CZ-613D…定価合計¥483,000▶**¥339,800**

| 12回 | ¥30,000 | 24回 | ¥15,900 | 36回 | ¥11,000 | 48回 | ¥ 8,700 | 60回 | ¥7,300 |
|---|---|---|---|---|---|---|---|---|---|

Ⓑ CZ-653C+CZ-613D…定価合計¥420,000▶**¥289,800**

| 12回 | ¥25,600 | 24回 | ¥13,600 | 36回 | ¥ 9,400 | 48回 | ¥ 7,400 | 60回 | ¥6,200 |
|---|---|---|---|---|---|---|---|---|---|

Ⓒ CZ-623C+CZ-613D…定価合計¥633,000▶**¥420,000**

| 12回 | ¥37,100 | 24回 | ¥19,600 | 36回 | ¥13,700 | 48回 | ¥10,700 | 60回 | ¥9,000 |
|---|---|---|---|---|---|---|---|---|---|

Ⓓ CZ-604C+CZ-606D…定価合計¥427,800▶**¥297,800**

| 12回 | ¥26,300 | 24回 | ¥14,000 | 36回 | ¥ 9,700 | 48回 | ¥ 7,600 | 60回 | ¥6,400 |
|---|---|---|---|---|---|---|---|---|---|

Ⓔ CZ-653C+CZ-606D…定価合計¥364,800▶**¥251,000**

| 12回 | ¥22,200 | 24回 | ¥11,800 | 36回 | ¥ 8,200 | 48回 | ¥ 6,400 | 60回 | ¥5,400 |
|---|---|---|---|---|---|---|---|---|---|

Ⓕ CZ-623C+CZ-606D…定価合計¥577,800▶**¥389,000**

| 12回 | ¥34,300 | 24回 | ¥18,200 | 36回 | ¥12,600 | 48回 | ¥ 9,900 | 60回 | ¥8,300 |
|---|---|---|---|---|---|---|---|---|---|

Ⓖ CZ-604C+CU-21HD‥定価合計¥496,000▶**¥352,000**

| 12回 | ¥31,100 | 24回 | ¥16,500 | 36回 | ¥11,400 | 48回 | ¥ 9,000 | 60回 | ¥7,600 |
|---|---|---|---|---|---|---|---|---|---|

Ⓗ CZ-653C+CU-21HD‥定価合計¥433,000▶**¥303,000**

| 12回 | ¥26,800 | 24回 | ¥14,200 | 36回 | ¥ 9,900 | 48回 | ¥ 7,700 | 60回 | ¥6,500 |
|---|---|---|---|---|---|---|---|---|---|

Ⓘ CZ-623C+CU-21HD‥定価合計¥646,000▶**¥438,000**

| 12回 | ¥38,600 | 24回 | ¥20,500 | 36回 | ¥14,200 | 48回 | ¥11,200 | 60回 | ¥9,400 |
|---|---|---|---|---|---|---|---|---|---|

★本体セットは、1ヶ月間だけの大特価セール!!
★クレジット価格は、消費税込みですョ。ご利用下さい!!

## X68000ソフト大セール実施中!!(ゲームソフト25〜30%OFF)　送料¥500

〈グラフィック〉●Z's STAFF PRO68K Ver.2.0
(シャフト)定価¥58,000
……………特価¥39,400

〈開発ツール〉●C-コンパラPRO68KV.2
定価¥44,800 CZ-245IS
……………特価¥33,300

〈データベース〉●CARD PRO68K Ver.2.0
定価¥29,800 CZ-253BS
……………特価¥22,300

〈グラフィック〉●C-TRACE+
定価¥198,000
…………特価¥145,000

〈C言語〉●C & Professional Pack
定価¥58,000
……………特価¥41,000

〈音楽〉●Music studio PRO68K Ver.2.0
定価¥28,800 CZ-261MS
……………特価¥21,500

〈CGツール〉●CANVAS PRO68K
定価¥29,800 CZ-249GS
……………特価¥22,200

〈ワープロ〉●Multiword PRO68K
定価¥32,000 CZ-225BS
……………特価¥23,800

〈通信〉●Tlepotion PRO68K
定価¥22,800 CZ-258BS
……………特価¥17,000

| 型名 | 商品 | 定価 | 特価 | 型名 | 商品 | 定価 | 特価 |
|---|---|---|---|---|---|---|---|
| CZ-212BS | BUSINESS PRO-68K | ¥ 68,000 | ¥ 48,000 | EW | | ¥ 38,000 | ¥ 29,800 |
| CZ-213MS | MUSIC PRO68K | ¥ 18,800 | ¥ 13,500 | G-68K | | ¥ 14,800 | ¥ 11,400 |
| CZ-214MS | SOUND PRO-68K | ¥ 15,800 | ¥ 11,500 | E-68 | | ¥ 19,800 | ¥ 15,300 |
| CZ-215MS | Sampling PRO-68K | ¥ 17,800 | ¥ 12,800 | CZ-255GS | CANVASドローグラフィックLIB | ¥ 8,800 | ¥ 29,600 |
| CZ-219SS | OS-9/X68000 | ¥ 29,800 | ¥ 21,000 | CZ-256GS | CANVASドローグラフィックVOL.2 | ¥ 8,800 | ¥ 6,600 |
| CZ-220BS | DATA PRO-68K | ¥ 58,000 | ¥ 41,000 | CZ-260LS | XBAS to CHECKER PRO68K | ¥ 9,800 | ¥ 6,600 |
| CZ-223CS | Communication PRO-68K | ¥ 19,800 | ¥ 14,300 | CZ-259SS | SX-WINDOW Ver.1.0 | ¥ 6,800 | ¥ 7,500 |
| CZ-224LS | THE 福袋 V2.0 | ¥ 9,900 | ¥ 7,500 | CZ-234LS | AI-68K | ¥188,000 | ¥ 5,000 |
| CZ-241BS | システム手帳リフィル集 | ¥ 9,800 | ¥ 7,500 | CZ-251BS | ハイパーワード | ¥ 39,800 | ¥139,000 |
| CZ-242BS | 活用フォーム集 | ¥ 9,800 | ¥ 7,500 | | KAMIKAZE | ¥ 68,000 | ¥ 29,800 |
| CZ-244SS | Homan 68K Ver2.0 | ¥ 9,800 | ¥ 7,500 | | デジタルクラフト | ¥ 39,800 | ¥ 45,400 |
| CZ-247MS | MUSIC PRO-68K〔MIDI〕 | ¥ 28,800 | ¥ 20,800 | | サイクロエキスプレス$\alpha$68 | ¥ 97,000 | ¥ 28,000 |
| CZ-240BS | Stationery PRO-68K | ¥ 14,800 | ¥ 11,500 | | C-TRACE 68 Ver3.0 | ¥ 78,000 | ¥ 73,000 |
| CZ-243BS | CYBER NOTE PRO-68K | ¥ 19,800 | ¥ 15,200 | | C-TRACE TP+ | ¥398,000 | ¥ 69,500 |

## 熱転写カラー漢字プリンター(ケーブル用紙付)　送料¥1,000

■CZ-8PC5 NEW

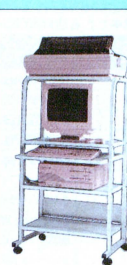
●48ドット
●熱転写カラー漢字プリンター
定価¥96,800
特価¥69,800

①CZ-8PK10(24ピン漢字プリンター136桁)
定価¥ 97,800……大特価!!TEL下さい。

②CZ-8PGI(24ピンカラー漢字プリンター80桁)
定価¥130,000……大特価!!TEL下さい。

③CZ-8PG2(24ピンカラー漢字プリンター136桁)
定価¥160,000……大特価!!TEL下さい。

③IO-735×(カラーイメージシェット)
定価¥248,000………大特価¥177,000

## パソコンラック〈送料無料〉

Ⓐ5段キャスター付
スライド式キーボード台
●1150(H)×640(W)×600(D)
定価¥38,000
特価 ¥15,000

Ⓑ4段キャスター付
●1250(H)×640(W)×700(D)
定価¥29,800
特価 ¥11,000

## 店頭新作ゲームソフト25〜30%OFF!!ビジネスソフト25%より特価中

**★通信販売お申込みのご案内★**　〒144 東京都大田区蒲田4-6-7 TEL:03-3730-6271

お申込みはお電話でお願いします。お客様の〈住所〉〈氏名〉〈電話番号〉及び〈商品名〉をお知らせ下さい。●入金確認後ただちに商品をご送付いたします。

**現金一括払い**
銀行振込:お近くの銀行より(電信扱い)にてお振込み下さい。
現金書留:封筒の中に住所・氏名・商品名をご記入の上当社までお送り下さい。

**クレジット**
専用お申込用紙をお送り致します。ので、必要事項をご記入、ご捺印の上ご返送下さい。手続きは簡単です。

オクト ラクラク クレジット表

| 3回 | 3.5 | 6回 | 4.5 | 10回 | 6.0 | 12回 | 6.0 |
|---|---|---|---|---|---|---|---|
| 15回 | 9.0 | 18回 | 11.0 | 20回 | 12.0 | 24回 | 12.5 |
| 30回 | 17.0 | 36回 | 17.5 | 48回 | 23.0 | 60回 | 33.0 |

**振込先**
富士銀行 久ヶ原(クガハラ)支店 当No.1824
三菱銀行 蒲田支店 当No.0278691
株式会社 億人(オクト)

※掲載の価格は変動しますので、まずは、お電話にてご確認ください。
※上記料金には、消費税は含まれておりません。消費税が付加されますので、詳しくは電話でお問合せ下さい。
※銀行振込、または、現金書留でご注文の際には、あらかじめ電話でご確認の上、お申し込み下さい。

各店のお休み
7月のお休み／4日(木)・11日(木)・18日(木)・25日(木)

# ワールドインアオヤマにおまかせ下さい！

## INFORMATION

**電話でのご注文の場合**

03-3987-7771

北海道受注センター 011-251-6771
九州受注センター 092-672-7771
お好きな時間にお電話を！

**ファクシミリでご利用の場合**

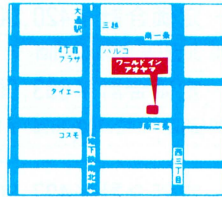
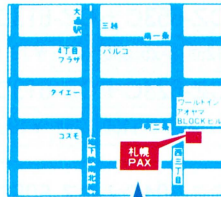
03-3985-5221

●ご注文方法（黒色のボールペン、またはサインペンでご記入下さい。）
(1)電話番号・住所・氏名又はお客様番号、お支払い方法をご記入下さい。

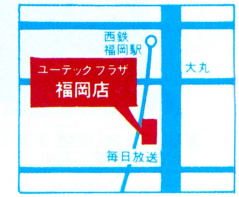
**お客様相談室**

03-3987-7795

すでにご注文いただいているお届け時間（時期）やメンテナンス、その他のお問い合せは上記へお電話下さい。

## 旭川店
旭川市4条8丁目ツジビル
■営業時間 11:00〜19:00

## 札幌店
札幌市中央区南2条西3丁目
リンクエギビル3F
■営業時間 11:00〜19:30

## 札幌PAX店
札幌市中央区南2条西2丁目
ブロックビル6F
■営業時間 11:00〜19:30

## 池袋店ソフト店
豊島区東池袋1-28-6
パールシティビル2F
■営業時間 11:00〜19:00

## 池袋本店
豊島区東池袋1-28-1
■営業時間 11:00〜19:00

## 福岡店
福岡市中央区渡辺通り4-9-25
ユーテックプラザ3F・地下鉄天神駅下車3分
■営業時間 11:00〜19:30

# X68000 新コース

SHARP

福岡店のお休み
7月のお休み／3日(水)・10日(水)・17日(水)・24日(水)
6月のお休み／5日(水)・12日(水)・19日(水)・26日(水)

### X68000EXPERT II X68000 Aコース
CZ-603C（本体）……………… ¥338,000
CZ-603D（0.31カラーディスプレー）……… ¥ 84,800
住友3M 5'2HDブランクディスケット…¥ 18,000
御希望ゲームソフト（人気ソフト上記よりお選び下さい。） サービス

定価合計 ¥440,800➡¥295,000

| | |
|---|---|
| ¥ 8,300×48回 | ㋭なし 頭なし |
| ¥15,200×24回 | ㋭なし 頭なし |

### X68000EV I X68000 Bコース
CZ634C TN（本体）……………… ¥368,000
CZ606D TN……………… ¥ 79,800
住友3M5'2HDブランクディスケット ¥ 18,000
御希望ゲームソフト……… サービス

合計¥465,800➡現金大特価
安すぎて表示できません。
クレジットでもお申し込みできます。

### X68000EV I X68000 Cコース
CZ634C TN……………… ¥368,000
CZ605D（Newタイプ）……… ¥ 99,800
住友3M5'2HDブランクディスケット ¥ 18,000
御希望ゲームソフト……… サービス

合計¥485,800➡現金大特価
安すぎて表示できません。
クレジットでもお申し込みできます。

### X68000EV I X68000 Dコース
CZ644C TN……………… ¥518,000
CZ606D TN……………… ¥ 79,800
住友3M5'2HDブランクディスケット ¥ 18,000
御希望ゲームソフト……… サービス

合計615,800➡現金大特価
安すぎて表示できません。
クレジットでもお申し込みできます。

### X68000EV I X68000 Eコース
CZ644C TN……………… ¥518,000
CZ605D（Newタイプ）……… ¥ 99,800
住友3M5'2HDブランクディスケット ¥ 18,000
御希望ゲームソフト……… サービス

合計635,800➡現金大特価
安すぎて表示できません。
クレジットでもお申し込みできます。

### X68000PRO II X68000 Fコース
CZ-653C（本体）……………… ¥285,000
CZ-602D（0.39カラーディスプレーテレビ）‥ ¥ 99,800
住友3M 5'2HDブランクディスケット…¥ 18,000
御希望ゲームソフト（人気ソフト上記よりお選び下さい。） サービス

定価合計 ¥402,800➡現金特価

| | |
|---|---|
| ¥ 7,200×48回 | ㋭なし 頭なし |
| ¥13,100×24回 | ㋭なし 頭なし |

### X68000PRO II X68000 Gコース
CZ-653C（本体）……………… ¥285,000
CZ-603D（0.31カラーディスプレー）……… ¥ 84,000
住友3M 5'2HDブランクディスケット…¥ 18,000
御希望ゲームソフト（人気ソフト上記よりお選び下さい。） サービス

定価合計 ¥387,800➡¥242,000

| | |
|---|---|
| ¥ 6,500×48回 | ㋭なし 頭なし |
| ¥11,900×24回 | ㋭なし 頭なし |

X68000をはじめソフト&周辺機器類は、当社池袋店・札幌店・旭川店・福岡店にて実演中です。各店X68000コーナーが常設されております。

## X68000ソフト&周辺機器

| | | | | | |
|---|---|---|---|---|---|
| SCSIボード（CZ-6BSI） | ¥ 29,800➡現金特価 | BF-68PRO | ¥ 15,500➡¥ 16,800 | Communication PRO-68K | ¥ 19,800➡現金特価 |
| システムサコム MIDIボード（SX-68M） | ¥ 19,800➡¥ 15,300 | CZ-6TU | ¥ 25,000➡現金特価 | Stationary PRO-68K | ¥ 14,800➡現金特価 |
| LANボード | ¥268,000➡¥201,000 | オムロンMD-24FP4 II | ¥ 38,800➡¥29,800 | DATA PRO-68K | ¥ 58,000➡¥ 43,500 |
| RS-232Cケーブル（平行） | ¥ 7,200➡現金特価 | オムロンMD-24FP5 II | ¥ 42,800➡¥ 33,000 | BUSINESS PRO-68K | ¥ 68,000➡¥ 51,000 |
| RS-232Cケーブル（クロス） | ¥ 7,200➡現金特価 | ローランドMT-32 | ¥ 64,000➡¥ 54,400 | NEW Printshop PRO-68K | ¥ 19,800➡現金特価 |
| インテリジェントコントローラ | ¥ 23,800➡¥ 18,900 | Hyperword | ¥ 39,800➡現金特価 | グラフィックライブラリ vol.1 | ¥ 8,800➡現金特価 |
| トラックボール | ¥ 13,800➡¥ 12,000 | CYBERNOTE PRO68K | ¥ 19,800➡現金特価 | グラフィックライブラリ vol.2 | ¥ 8,800➡現金特価 |
| ジョイカード（延長コード付） | ¥ 3,200➡¥ 2,900 | C compiler PRO-68K | ¥ 44,800➡¥ 33,600 | Musicstudio PRO-68K ver1.1 | ¥ 28,800➡¥ 21,600 |
| CZ-8BS1（X-1用） | ¥ 23,800➡現金特価 | CARD PRO-68K | ¥ 29,800➡現金特価 | MUSIC PRO-68K〔MIDI〕 | ¥ 20,500➡現金特価 |
| 拡張I/Oボックス | ¥ 88,000➡現金特価 | CARD PRO システム手帳リフィル集 | ¥ 9,800➡現金特価 | ソングライブラリ —101曲集— | ¥ 8,800➡現金特価 |
| アンプ内蔵スピーカーシステム | ¥ 28,500➡現金特価 | CARD PRO 活用フォーム集 | ¥ 9,800➡現金特価 | Sampling PRO-68K | ¥ 12,500➡現金特価 |
| システムラック | ¥ 44,800➡¥ 35,800 | SX-WINDOW ver1.0 | ¥ 6,800➡現金特価 | SOUND PRO-68K | ¥ 15,800➡¥ 11,500 |

## X68000シリーズ周辺機器

| | | | | | |
|---|---|---|---|---|---|
| CZ-8NS1 | ¥188,000➡¥141,000 | CZ-8PC5 | ¥ 94,800➡現金特価 | I/Oデータ 2MB増設RAM | ¥ 50,000➡¥ 36,500 |
| CZ-6BN1 | ¥ 29,800➡現金特価 | IO-735X | ¥248,000➡現金特価 | I/Oデータ 4MB増設RAM | ¥ 88,000➡¥ 64,000 |
| CZ-6VT1 | ¥ 69,800➡¥ 52,400 | CZ-8PK10 | ¥ 97,800➡現金特価 | GP-IBボード | ¥ 59,800➡現金特価 |
| CZ-6BV1 | ¥ 21,000➡現金特価 | 1MB増設RAM（CZ-600C専用） | ¥ 35,000➡¥ 28,000 | 増設用RS-232Cボード | ¥49,800 ➡現金特価 |
| CZ-6PV1 | ¥198,000➡¥148,500 | 1MB増設RAM | ¥ 28,000➡¥ 22,400 | ユニバーサルI/Oボード | ¥ 39,800➡現金特価 |
| CZ-8PC3 | ¥ 65,800➡¥ 39,000 | 2MB増設RAM | ¥ 60,000➡現金特価 | 数値演算プロセッサ | ¥ 61,000➡現金特価 |
| CZ-8PG1 | ¥130,000➡¥ 97,500 | 4MB増設RAM | ¥138,000➡¥107,000 | FAXボード | ¥ 79,800➡¥ 55,800 |
| CZ-8PG2 | ¥160,000➡現金特価 | I/Oデータ 1MB増設RAM | ¥ 25,000➡¥ 18,000 | MIDIボード | ¥ 26,800➡¥ 20,300 |

## X68000万全のサポート

AOYAMAにて購入のX68000は万一故障の場合でも全国どこでも出張サービスがうかがいます。 万一の場合ワールドインアオヤマサポート係にお電話下さい。お客様のお名前と電話番号だけで手続きは完了。

| 組合せ自由 | 激安金利にキャンパスクレジット | ゆっくり、お支払いは8ヵ月先から |
|---|---|---|
| 各コース以外の組合せもコースをベースに周辺を合せたセット……お支払いだって御希望のパターンをお組みいたします さあ、ご相談もお見積りも受注センターもしくは各店へお気軽に | 手続きカンタン、大学生の為の超低金利クレジット。20歳以上の学生の方は原則として保証人様には連絡いたしません | クレジット業界最低の金利を有効に使って、支払いは最長8ヵ月後から始まるクレジットでも。 |

## これは？と思ったらどんどんお電話下さい！

### グレー限定お買得セット X68000 新Uコース
CZ-603CGY（本体）……………… ¥338,000
CZ-606D（カラーディスプレー）……… ¥ 79,800
住友3M2HDブランクディスケット……… ¥ 18,000
御希望ゲームソフト（上記ソフトよりお選び下さい） ¥サービス

定価合計 ¥450,800➡¥295,000

| | |
|---|---|
| ¥ 6,100×72回 | ㋭なし 頭なし |
| ¥ 6,900×60回 | ㋭なし 頭なし |
| ¥10,300×36回 | ㋭なし 頭なし |
| ¥14,900×24回 | ㋭なし 頭なし |

### 一歩ふみこんだアートの世界 X68000 アール Rコース
CZ.604C（本体）……………… ¥348,000
CZ-602D（0.39カラーディスプレー付ディスプレー）…¥ 99,800
住友3M2HD ブランクディスケット……¥ 18,000
御希望ゲームソフト（人気ソフト上記よりお選び下さい） ¥サービス

定価合計 ¥465,000➡¥316,000

| | |
|---|---|
| ¥ 2,800×60回 | ㋭¥30,000 頭なし |
| ¥ 5,000×36回 | ㋭¥40,000 頭なし |
| ¥ 7,800×60回 | ㋭なし 頭なし |
| ¥11,900×24回 | ㋭¥30,000 頭なし |

### X68000お買上げのお客様へ
各コースで御希望ソフトは「サンダーブレード」「ダウンタウン熱血物語」「ニュージーランドストーリー」「沙羅曼蛇」「ツインビー」「フルスロットル」「パックマニア」「ビーチバレー」「アルカノイド」「熱血高校ドッチボール」のうちいずれかからお選び下さい。

SOFT BANK

スーパーファミコンまるかじり!

# The スーパーファミコン

第12号(6/28号)

付録
きっと役立つスーパー読本
スーパーウルトラベースボール&スーパープロフェッショナルベースボール

特集
## どれがお得かよーく考えてみよう! PART.2
今年前半に発売されたソフトを総チェックなのだ。

すぎやまこういちのゲーム漂流記
第8回ゲスト
中村勘九郎

読んで得するスーパーガイド
㊢新作SUPERGUIDE
ワンダラーズ フロム イース
ガデュリン/ドラッケン

好評発売中
定価380円(税込)
隔週金曜日発売

# BEEP! POWERFUL MEGA-MAGAZINE MEGADRIVE

ビープ!
▶メガドライブ◀ 7月号

好評発売中
定価480円(税込)
毎月8日発売

特集 セガ CD-ROM発表!
今秋発売予定のニューマシンの速報

ALL ABOUT ソニック・ザ・ヘッジホッグ キャラクターゲームのニューウェイブを探る

新作▶アドバンスド大戦略/レッスルウォー/マーベルランド/空牙 ほか

特別2大付録
ファステストワン攻略ガイド&
ソニック(B5判)ステッカー

■最寄りの書店でお早めにお買い求めください ソフトバンク出版事業部

# シャープパソコンフォーラム'91

▲池袋にあるサンシャイン文化会館で行われたパソコンフォーラム。写真左隅に見える水色の紙袋を受付でもらいます。X68000のパンフレットなどが入っていました。それではいざ会場へ。

◀会場に入った僕たちを待ち受けていたのは，XVIの看板も目に入らないくらいに派手な柱にディスプレイが4つ。でも，よく見るとXVIは置いてないんですね。疑り深い人は確かめたかもしれませんが，ケーブルをたどったらちゃんとXVIにたどりつきました。写真の右隅に見えるのは記念品の交換所。アンケートに答えて，X68000特製シャープペンシルをもらいました。というわけで，左側に入っていきましょう。

▲左側を向くとCGAが。本誌でもお馴染みのDŌGAです。メモリは12Mバイトのフル実装。1分程度のアニメを流していました。21インチのディスプレイは迫力モノです。ちなみに光磁気ディスクは参考出品だとか。

◀特設イベント会場では山下章氏の司会によるユーザー自作ソフトの発表会や本誌編集長の講演会などが行われました。見てのとおりの賑わいぶり。あらためてX68000ユーザーのパワーを感じる次第です。

▲入り口の裏側ということは背景でわかりますね。中央にデ〜ンと置いてあるのがXVI。入り口の4台のディスプレイとあわせて9台のディスプレイがこのXVIにつながっていました。コンパニオンのお姉さまは桜田淳子のモノまねをして……るわけではありません。

◀X68000シリーズはXVIしか展示されていませんでした。縦型コンピュータ全盛の会場でここのAXコーナーにあるパソコンたちはちょっと浮いていましたね。このブースを熱心に見ている人たちはちょっと大人の人でした。

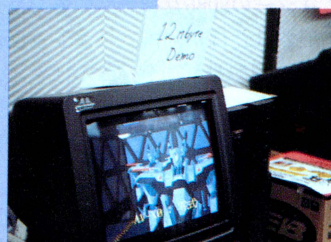
▲X68000オリジナルグッズの販売もありました。フロッピータイトルシールなどが人気だったようです。また，電飾ポップなども売れていました。ネクタイピンなどは年齢層が若いせいかイマイチだったとか。そうそう，土曜日の会場でX68000ポーチを拾った人は編集部の山田までご一報ください。

▲ワールドインアオヤマがソフトなどの即売をやっていました。なんと，V'BALLとニュージーランドストーリーが税込2,000円で売っていました。このチャンスにB級名作を手に入れた人も多かったんじゃないかな。もちろん，マニュアル付きパッケージ入りの新品ソフトでした。

▶写真では片側しか見えませんが，両サイドにXVIがズラリと並んでいました。もちろん見て触って16MHzを堪能できます。ソフトはMultiword PRO-68K，CARD PRO-68K ver.2.0，Musicstudio PRO-68K，SX-WINDOW Ver1.1などがありました。

▶ステゴちゃんは健在です。キャストのブースではトランスピュータのカタログなどがもらえました。ステゴちゃんの即売会はないのかなぁ。

▲ズームのブースは人だかり。アンケートに答えると，ファランクスのポスターと歴代エンベローブ3枚組がもらえました。本紙1990年度GAME OF THE YEAR助演キャラクター賞のネコはこの会場でも人気抜群。特製の紙袋に入れてもらえて大満足です。

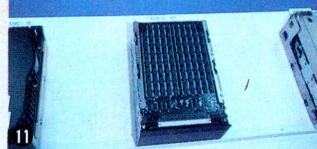
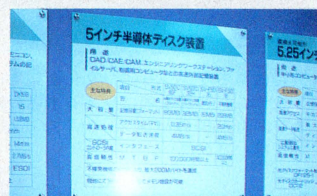
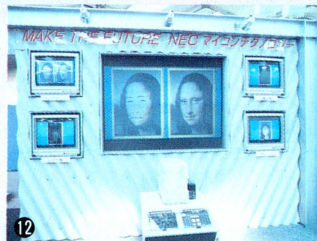
▶会場全体は若い活気にあふれかえり，非常に盛況です。目当ての場所があってもたどりつくまでがひと苦労。ほんとに皆さんお疲れさまです。

◀ちょっとダークサイドな映像。心霊写真の一種ということにしておきましょう。XVIを使って10MHzで動かす余裕ぶり。12Mバイトとありますが，使い切ってない気もしますね。なお，この件に関してのお問い合わせはご遠慮ください。

## X68000XVI勢揃い

X68000XVIのお披露目として，5月11，12日にシャープパソコンフォーラム'91が行われました。場所は池袋のサンシャイン文化会館。ついに現れた16MHzの手応えを確かめるためにEXE会員のみならず，これからX68000ユーザーになろうとするものまで山のような人だかり。

会場ではXVIの発表のみならず，各ソフトハウスの新作ソフトの展示にゲーム大会，ユーザー自作ソフトの発表会，ソフト/ハードの即売会など盛り沢山の内容。賑やかに2日間となったようです。

なお，レポーターはS.K.君にお願いしました。

# マイクロコンピュータショウ'91

❶会場風景。
❷シャープブースだ。
❸X68000XVI，ソフトはEasypaint SX-68K，ダッシュ野郎など。
❹本体は見えないが，X68000による解説デモ。グラフィック画面に絵を読んでいるだけという話もある。
❺大型，高解像度の白液晶。
❻十分な大きさと発色性のカラー液晶。普及は目前か？
❼データフロー型プロセッサの3D CGへの応用例。
❽セガブースのテラドライブ。
❾ちなみに，これがテラドライブの中身だ。
❿日立マクセルの半導体ディスク。これは80Mバイトだが200Mバイトまでラインアップされている。
⓫アーケード版のストライクイーグルII(海外版）ではAMDの32ビットRISCチップを採用している。
⓬国産32ビットCPU, V70を使った画像処理のデモ。
⓭画像認識を応用したヨットの自動制御。

## マイコンショウ

5月8～11日の4日間マイクロコンピュータショウ'91が例年どおり東京流通センターで開催された。

「マイクロコンピュータショウ」とはいうものの，展示内容は部品や基礎技術，開発機器関係が主体となっている。ここで発表された技術が何年かたつとパソコンでもお馴染みになっていることもある。業界の将来を占ううえではやはり欠かせない催し物であろう。

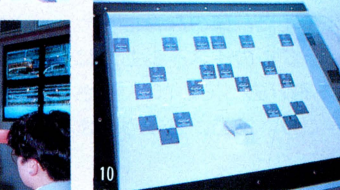
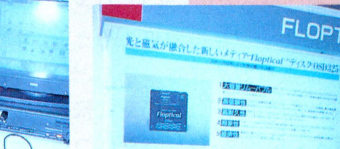
全体としてワークステーションの進出が目立ち，今年の傾向として「ファジィ開発システム」などが目を引く。CPUはRISCへ，組み込み制御用マイコンも画像認識などの要求の高度化に伴い，ハイビット化が進みつつある。

さて，恒例（？）のTRON協会は例年ほどのスペースではなかったものの，いわゆるTRONチップ関係の各社は32ビットCPUを展示など元気な印象を受けた。

そのほか，VMテクノロジーの8086互換チップも32ビット版が出展されたほか，国産CPUでは先行していた日本電気はV70を中心に画像処理のデモを展開して高性能をアピールしていた。

シャープ関係ではX68000XVIとEasypaint SX-68Kや新しいSX-WINDOWの展示を始め，データ駆動型プロセッサの適用範囲の広さと高性能をアピールするデモのほか，パソコンレベルでスーパーコンピュータ並みの画像処理を行うための画像処理プロセッサなどを展示。会場でのビンゴ大会ほか，半導体関連製品のデモにも（本体は見えなかったものの）X68000が使われていたことが目を引いた。

そのほか，毎年マイコンショウで新作ソフトウェアを発表する電波新聞社では，今年はタイトーのアーケードゲームからの移植作品，キャメルトライを初公開。見た感じでは画面全体の回転もそつなくこなしており，はっきりデモバージョンと書いてあったものの，市販されるのもそう遠くなさそうな気配だった。

# 第72回ビジネスショウ

❶会場となった晴海国際見本市会場。
❷これがフルカラーファクシミリ。
❸X68000とスキャナ，カラープリンタを使ったデモ。
❹All in NOTEのVGAバージョン。
❺ついに発売された壁掛けテレビ，500,000円。ちなみに画面は9インチ。
❻標準となるか？　IBMの3.5インチ光磁気ディスクドライブ。
❼「IBM」でのテラドライブ。
❽おや，AMIGA2000と一部で噂のビデオトースターがこんなところに。
❾どこかで見たよなマシン。Q2回線制御で大活躍とか。
❿マクセルの「フロプティカル」ディスク。3.5インチサイズのメディアと小型ドライブで1枚あたり25Mバイトの容量。
⓫CD-I元年となるか？　これはソニーのCD-I。液晶テレビつきでポータブル。
⓬ローランドDGのパーソナルカッティングマシン，STIKA。
⓭地球に優しい再生紙の文房具。

## ビジネスショウ

5月15～18日の4日間，晴海国際見本市会場で第72回ビジネスショウが開催された。

これまでもビジネスショウの出展品の中心的なテーマでありながらどうも現実味のなかったネットワーク関係の出品が急速に身近なものとして姿を見せていた。これはNetware，LAN Managerらの登場によるものだろう。大型サーバや専用システムではなく，パソコンと日常的な環境がネットワークにつながることの意味は大きい。これからのコンピュータの方向性を大きく左右することになると思われる。

そして新しい標準となる可能性を持つDOS/Vマシンも数社から出品され注目を集めていた。これまでPC-9801市場を守っていた「日本語」の壁が崩されようとしているのだ。ソフトではまだ疑問が残るものの，PC関連の完成されたハードウェアがそのまま使えるだけでもメリットはあろう。

今年は新しいOSが目白押し。DOS/VやWIN3はいうに及ばず，OS/2 ver.2.0，Macintosh system7.0，日本語NextStep，そしてCD-Iまで現れた。

シャープブースではOAのカラー化を全面に押し出した内容。特に昨年は参考出品だったカラーファクシミリ。ファクシミリは画像が汚いという常識を覆し，カラー原稿とFAXを並べ「どちらがオリジナルか？」とくる大胆さ。実際それくらい綺麗でほとんど見分けはつかない。さらにX68000とイメージスキャナ，カラープリンタを組み合わせてイメージ処理を展開。

そして，壁掛け液晶テレビだ。もう何年も前から出品されているものだが，今回は500,000円という定価がついているのだ。エレクトロニクスショウやマイコンショウで参考出品されているのとは違って，ここはビジネスショウ会場だ。ついに壁掛けテレビが商品化されたかと思うと感慨もひとしお。

昨年も女性客が多いと書いた覚えがあるが，今回は特にOLが多かった。さらに社会見学の一環としてか制服姿の女子高生の群れまで……。なんだこれは？　(S.N.)

[第2回]

進化しつづける画材

寺尾 響子

# 響子inCGわ〜るど

しみだらけで，机からはみ出すほど大きな製図板に，大学ノートくらいの紙をおきます。鼻の先がつくほど紙に顔を近づけて描きこんでいく……それが，私の仕事です。さきほど，8時間かかってビーグル犬のイラストをエアブラシとペンで仕上げました。明日の納期にも間に合ったし，さて，ひとやすみ。コーヒーが入ったカップを製図板のうえにおき，出来上がった作品をながめます。こんどは，このビーグルをセスナ機に乗せて，蒼い空に飛ばそうかしら，などと考えています。ひじがすべって，カップにあたりました。覆水，盆に返らず。茶色に染まったイラストをまえに，頭のなかは真っ白になります。新しいビーグルのイメージもなにもかも，セスナ機もろとも飛び去ってしまい，残されたのは「締切」の2文字だけ。ああ，きょうもまた徹夜です。

わたしは
すべての肉体的作業から解放されて
イメージのみの作品を
ひとりになって
つくりたかった
それには
パーソナルなコンピュータによる
CGしかなかった

KYOKO

## CGという画材

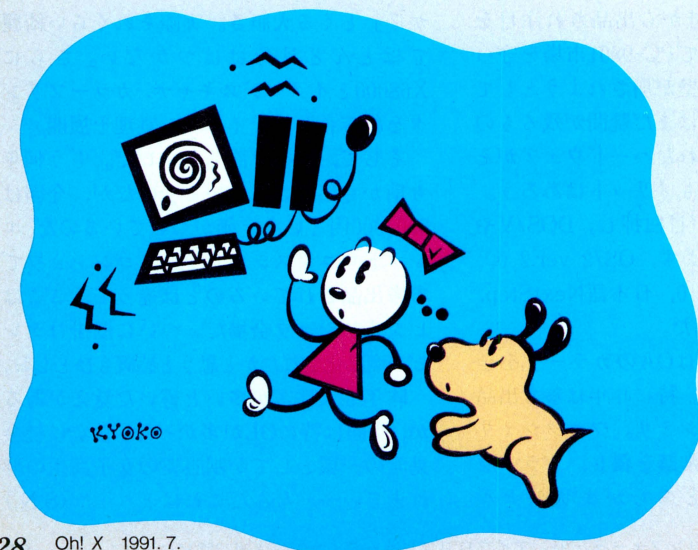
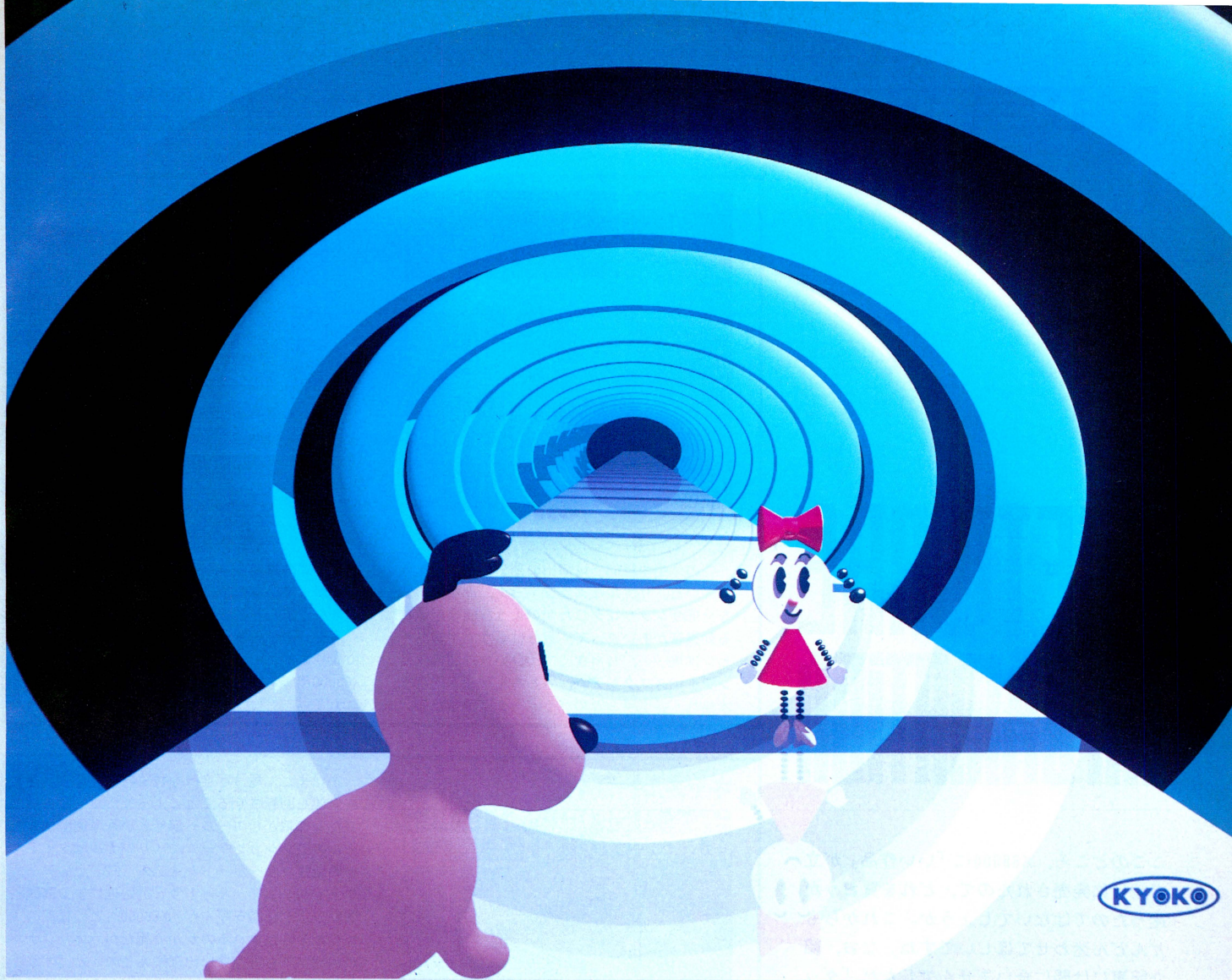
現実にはありえない構造物や動植物が，ぽっかりと浮かんでは消えていきます。私は自分の脳のなかにあるイメージが，どうやら空間的なものらしいとわかっていました。空間を平面に，わりと忠実に表現する絵画技法が遠近法[1]です。この技法を用いて自分のイメージを紙に定着するのはとても根気のいる作業です。また，絵を描いている間は，新たに浮かんだ構想を吟味することはできません。描くのに集中していないと線がゆがんだり，色がはみ出したりしてしまうからです。失敗したならば，はじめから描き直さなければなりません。できることなら，描く作業から肉体的にも時間的にも解放されて，自分自身のなかのイメージの世界を広げたり深めたりしたい，といつも思っていました。

そんなときに出会ったのがパソコンによる3次元CGでした。コンピュータが，3次元空間のデータを2次元に変換して，平面のCRT上に，色や明暗までつけて投影してくれるのです。画像が気に入らなければ，データを直して入力するだけで済みます。オリジナルとまったく同じコピーをつくるのも簡単です。さらに，いままで描く作業に取られていた時間を，景山民夫の本を読む，ぼんやり過ごす，手抜きでない夕食をつくる，などに使うことができます。パソコンCGの出現によって，CGも画材として扱える時代がやってきました。

いくらパソコンでできるといったって，画材にしては値段が高すぎるんじゃないか，と思う人もいるでしょう。でも，コンピュータのなかには，無尽蔵に紙も絵の具もあります。失敗して捨てるものは，まとわりつく未練と電気代だけなのです。

## 最小限のシステム

画材はまず基本セットから揃えましょう，ということで最小限のCGシステムを考えてみました。

まず，ハードウェア。X68000シリーズのパソコン本体，標準添付のキーボードとマウス。専用ディスプレイ。メインメモリは2Mバイトあれば十分です[2]。ハードディスクや数値演算プロセッサ，プ

リンタ，スキャナはとりあえず必要ありません[3]。もの足りなくなったらシステムアップすればよいと思います。

つぎにソフトウエア。使える３次元CGのソフトウエアは４つあります。このうちどれかひとつあれば作品はつくれます。C-TRACE68ver3.0，サイクロンExpressα68，Z'sTRIPHONY DIGITAL CRAFT，DōGA CGA SYSTEM[4]。それぞれに長所と短所があり，細かく挙げるときりがありません。違いをひと言でいうならば，「前の２つは，画質は高いがレンダリングに時間がかかる。あとの２つは画質は多少落ちるが，レンダリングが速い」ということになるでしょう。また，マッピングデータや背景などを作成するのに，Z'sSTAFF FRO-68Kがあると便利です。プログラミングの知識はとくにいりません。が，Human68kのユーザーズマニュアルはよく理解しておくと，自分に合った快適な制作環境が整えられます。

ハードウェアもソフトウエアもバージョンアップしたり，高性能で低価格な新しいものがつぎつぎと出てきます。追いついていけない，いま買ってもどうせ古くなる，と悲観的にみる向きもあります。でも，私にとってはうれしいのです。自分のイメージをもっと自由に表現できるようになると思えるから。絵を描く人間にとってCGはまさに「進化しつづける画材であり，技法」なのです。

---

1）遠近法（一点消失図法）を正確に用いて最初に絵画を描いたのは，初期ルネサンス期のマザッチオという画家でした。約500年も前のことです。この方法は平行線の処理に優れています。おもな定理は，画面を平面とすると，

　１．画面に垂直な平行線は，画面中央の１点（消点）で交わる
　２．画面に平行な平行線は，平行線として描かれる
　３．画面に斜めな平行線は，その傾きに応じて消点からずれた点で交わる

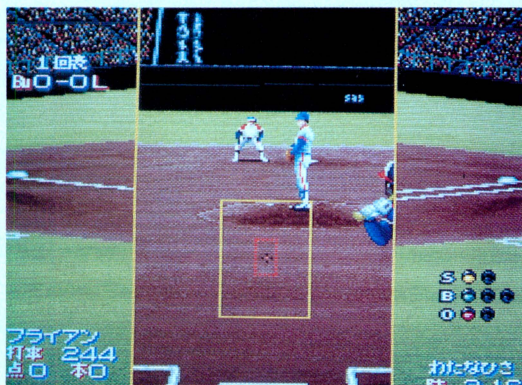
です。このほかいろいろな定理を駆使すると複雑な立体像を紙に写しとることができるのです。
ルネサンス以降，20世紀初頭にフォビズムとキュビズムが出現するまで，明暗法と遠近法は絵画の絶対要素となり，空間や物を再現する技量がそのまま画家の才能とされていました。

2）ソフトウェアによっては，マッピングデータをメモリにできるだけ読み込むとレンダリングのスピードが速くなるものがありますし，メモリ上にRAMディスクを作りワークファイルをおくとモデリングが速く行えるものもあります。レンダリングとはコンピュータが画像を描くこと，マッピングとは物体に模様をつけること，モデリングとは物体をデザインすることです。

3）トランスピュータもいりません。私も今は仕事で必要上使っていますが，ないときでも作品をつくっていました。なければないでなんとかなるものです。ただ，新しいFLOAT２は，システムに組み込むと，どのソフトウェアでもレンダリングが速くなるので入手したいものです。

4）DōGA CGA SYSTEMは，アニメーションを目的としたシェアウェア，パブリック・ドメイン・ソフト（PDS）の一種。ほかの３つは市販されています。

# SOFTWARE INFORMATION

ここのところ，X68000に「いい作品」が立て続けに発売されたので，どれを買おうか迷ったのではないでしょうか。これからもどんどん迷わせてほしいですね。なお，紹介記事には間に合いませんでしたが，「ダッシュ野郎」，「ライヒスリッター」，「サイレントメビウス」，「ドラゴンウォーズ」が発売中です。次号で取り上げるつもりですので，どうかお待ちを。

## 生中継68

この「生中継68」は演出の部分に相当力を入れている。リアルなグラフィック，球場内の音を全部サンプリングしたかのような臨場感溢れる効果音があいまって，気分はもう旭川スタルヒン球場……，じゃなくて，ひいき球団のフランチャイズスタジアム。もちろんゲーム内容も本格的。各選手ごとのパラメータは豊富だし，試合形式の設定もいろいろ。試合前後も面白く仕上がっているので，あとは肝心な部分のバランスをうまくとれるかどうかにかかっている。

注意，選手名をカタカナで入れると外人選手になります。（R.A.）

X68000用　5"2HD版2枚組　9,800円(税別)
コナミ エンタテイメント　☎03(3264)5678

## キャメルトライ

またまた，電波が度肝を抜く新作を発表。迷路全体を回転させせることによって，ボールを壁づたいに転がしゴールへ導くという単純明快ルールのアクションゲーム。オリジナルはタイトーの新鋭マザーボードシステム“F2”上のアーケードゲームで，ハードウェア処理による高速回転表示が話題を呼んだ。X68000へどの程度のレベルで移植されるのかが心配だが，オリジナルと比べても遜色ない出来栄えとのこと。オリジナルはパドルを使った操作だったので，それをどう再現するのかも注目される。（善）

X68000用　5"2HD版　価格未定
電波新聞社　☎03(3445)6111

### 再登場5作品の秘密とは?

| 順位 | タイトル | (前回順位) |
|---|---|---|
| 1. | パロディウスだ！ | 1 |
| 2. | 遙かなるオーガスタ | 2 |
| 3. | ファランクス | ー再 |
| 4. | A列車で行こうIII | 10↑ |
| 5. | ボンバーマン | ー再 |
| 6. | イース | ー再 |
| 7. | マーブル・マッドネス | 5↓ |
| 8. | 黄金の羅針盤 | ー初 |
| 9. | ソルフィース | ー再 |
| 10. | 三國志II | ー再 |

ますます強さに磨きをかけつつある「パロディウスだ！」。オーガスタ，ファランクスといった強豪をものともせずに突っ走っています。この上位3作に票が集中したために，下位集団では票数差が縮まって混戦になりました。ソルフィースや三國志IIなどがこの機に乗じて（?）再登場をはたしています。

ファランクスは発売のニュースが流れたときにランクインして以来ですから，発売後としてはこれが初めて。ビジュアル，デモ，ゲームバランスの評価は高いんですが，音楽については賛否両論というところですか。でも，「パロディウスだ！」というシューティング同士の競合がありながら3位までつけたのは立派。この先の伸びに期待しましょう。

A列車で行こうIIIも発売されると同時に6ランクのアップ。箱庭ゲームとしての面白さ，鉄道を走らせる楽しみなどがうまく組み合わされているのが人気の秘密。鉄道ファンが特にメロメロになっているようです。

5位にはボンバーマンがランクイン。小粒なゲームですが，対戦の面白さを買われてがんばっています。考えてみれば大勢で遊べるゲームのランクインもひさしぶり。長くチャートを賑わせてほしいものです。

初登場はリバーヒルの黄金の羅針盤のみになっています。推理アドベンチャーの大御所がひさびさに作品を発売するということで，熱い期待を集めているようですね。

というわけで，下位の混戦があわさって5作も再登場が出た今月のチャートでした。では来月までご免。（浦）

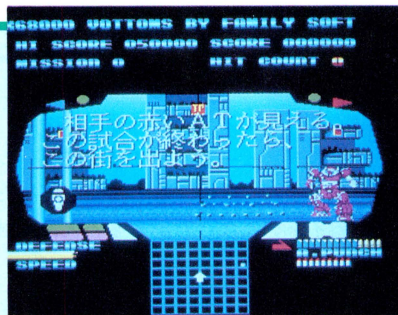
## 装甲騎兵ボトムズ DEAD ASH

「装甲騎兵ボトムズ」とは，メカもののなかでもリアリティや泥臭さを競わせたら右に出るものはないとまでいわれたアニメだ。

そのボトムズの世界が3Dシューティングゲームになった。うなるマシンガン，飛び交うミサイル，なんといっても真髄はアームパンチ。アーマードトルーパー（AT）を縦横無尽に走らせるローラーダッシュももちろんできる。夢にまで見たロボットアクションである。

主人公の名前はアーサー・ライトン。キリコ・キュービーでないのは残念だが，キャラクター設定や原画はプロのアニメーター加瀬政広氏が担当。

敵の包囲網を突破するのが目的で，高速強制スクロール面なども用意されている。ATに乗り込むキャラクターを選び，武器を選択すれば，もうそこはボトムズの世界。

ローラーダッシュ！　迷うことはない。迷えばそこには死があるのみだ。（S.K.）

X68000用　5″2HD版3枚組　8,800円（税別）
ファミリーソフト　☎03（3924）5727

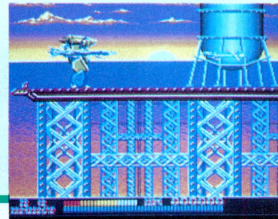
## オルテウスII

PC-88VA専用で通信販売されていたシューティングゲーム「オルテウス」がバージョンアップして登場する。敵も大きいけど，自分も負けず劣らず大きくて，画面の迫力はものすごい。この自機が画面いっぱいに6種類の特殊ショットをぶっぱなすさまは壮観だ。ジェネアス軍のなぞの超兵器の前に陥落してしまったオルテウス星。この星を救うため，ミリア・ランバート軍曹が失われた先史文明の遺産“ユニットX”を駆って，単身ジェネアス軍に立ち向かう。現在，αバージョンが届いているだけなので細かいところはお伝えできないが，経験値や金のシステムを採用しており，レベルアップによって耐久力を増やしたり，お金を貯めて兵器をパワーアップするというRPGのようなシステムも取り入れているということだ。制作元のウインキーソフトはX68000には今回が初のチャレンジ。その実力のほどに注目だ。（浦）

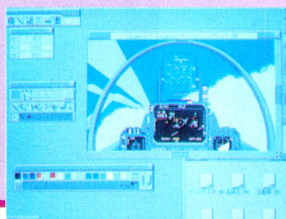
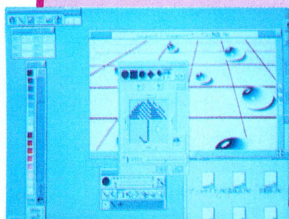
X68000用　5″2HD版　価格未定
ブラザー工業（TAKERU）　☎052（824）2493

## アクアレス

「ナイアス」で一躍その名を世にとどろかせたエグザクトの新作，「アクアレス」の開発途中画面が届きましたので，さっそく紹介いたしましょう。「深海38000m，静寂なる暗闇に幻想のシンフォニーが響きわたる！　群を抜くゲームシステムとバックストーリー！　本格派ロボットアクション登場！」と気合いの入った売り文句。なかなかの力作に仕上がりそうです（色づかいがあいかわらず「ナイアス」してますが）。リアルなロボットのほかに，デモにはアニメ調のキャラクターも登場するようですし，いやがおうにも期待が高まってしまいますよね。ラスタスクロールなど，エグザクト得意の特殊効果も健在のようですし，どこまで“魅せてくれる”のか，完成版を楽しみにしたいと思います。8月上旬発売予定でまだ画面写真のサンプルのみですが，近々また詳しくレポートできると思いますので，皆様お楽しみに！（哲）

X68000用　5″2HD版　価格未定
エグザクト　☎025（247）9160

## Easypaint SX-68K

SX-WINDOW用アプリケーション「Easypaint」が登場した。作図ウィンドウごとに65536色中16色のカラーを割り当てることができ，8種類のツールと8通りのモードをサポートしている。名前のわりには，絵を描くための道具はひと通り揃っているし，スキャナユーティリティの「Easyscan.X」とイメージ印刷ユーティリティの「Easyprint.X」も付属していて，結構本格的なツールに仕上がっている。画面上に開かれるウィンドウを操作し，ツールアイコンをクリックしながらそれぞれの機能を使っていくわけだが，マニュアルもかなり親切に解説されており，ふだんあまりSX-WINDOWを使っていない人や初心者でも十分使いこなせるようになっていて好感が持てる。しかし，アクティブウィンドウを切り替えるとツールウィンドウが閉じてしまうなど，ウィンドウアプリケーションとしてはものたりなさを感じてしまう面もある。（M.H.）

X68000用　5″2HD版　12,800円（税別）
シャープ　☎03（3260）1161

「Easypaint SX-68K」と同時に，「SX-WINDOW ver.1.1」も9,800円（税別）で発売されています。また，「Easypaint SX-68K」は「SX-WINDOW ver.1.1」上で動くアプリケーションですので，「SX-WINDOW ver.1.02」をお持ちの方は「ver.1.1」へのバージョンアップサービスを受けたうえでご利用ください。

お知らせ

### 「シューティング68K」コンテスト

今月号の“THE SOFTOUCH”でも紹介している「シューティング68K」。開発元のアモルファスではこの「シューティング68K」で作られたゲームのコンテストを開催します。

賞金のほうも，

| | | |
|---|---|---|
| 優勝 | 500,000円 | 1名 |
| 準優勝 | 200,000円 | 1名 |
| 入賞 | 100,000円 | 数名 |

と豪華なものになっていますので，皆さんも応募してみてはいかが。締め切りは8月31日。

### 目指せ，100万人

イマジニアでは「シムシティー　テレインエディター」を使用して，人口をどれだけ増やせるかを競うコンテストを開催します。いちばん人口の多かった読者には，平成3年9月1日から平成5年2月28日までの間に発売されるイマジニアのX68000用商品をプレゼント（特別モニターとして）。応募者は住所，氏名，年齢，職業，使用機種，達成人口を明記した書面を同封のうえ，データディスクをOh!X編集部「シムシティー大会」係まで郵送のこと。

# 大人のためのファランクス入門

Nishikawa Zenji
西川 善司

あの「ジェノサイド」でデビューし，その後「ラグーン」の発売で話題を呼んだズーム。新作は基本に立ち戻った(?)横スクロールシューティングになった。もちろん，手強い内容になっているぞ。

どうも最近くしゃみをしたようなネーミングが流行っているみたい。

へぇ，へぇ，……エクシヴィ
ふぁ，ふぁ，……ファランクス
へぇ，へ，……eXOn
ふ，ふ，……ループス
そ，そ，……ソルフィース
す，す，……スコルピウス
ふぉ，ふぉ，……FOXY2

と，まあ，こんな感じ。探せばまだまだありそう。さあ，君も意表を突いたくしゃみをしてみんなの人気者になろう。ま，ま，……マジカルショット。いやーん。

## 横スクは厳しいーのれーす◆◆◆◆◆◆

最近は横スクロールシューティングなんてものはちっとも珍しくない。コードレスホンやJRの冷房車両みたいなもんで，一般常識同然ともなっている。ゲームセンターに足を運べばあるわあるわ，横スクロールシューティングが。「グラディウス」もどきや「R-TYPE」もどき，「ダライアス」もどき……。薹(とう)の立ったじいさんなんかには絶対区別のつかない世界だろう。

あと，最近は拡大縮小などの特殊技術も常識化していてちょっとやそっとじゃ最近の小僧たちを驚かすことは無理である。よくプログラムをしたこともないような半ズボン姿の鼻水垂らした小僧が平然と「あ，ラスタスクロールじゃん」などと「通」ぶった口をきいている。

まあ，そんなわけでこの「横スクロールシューティング」で人に「おおぉ」といわせるには現代はとても難しい世の中なのである。

## ラグーンの雪辱をファランクス◆◆◆

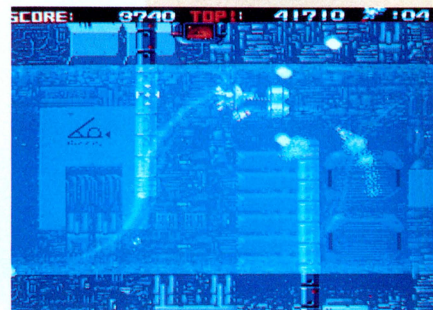
ま，そういうわけで「ファランクス」とのご対面のときにも「ちっとやそっとじゃ，驚きませんですわよ」といった心構えで臨んだのだが，そんな心構えは「ズームネコIPL」を見たとたん波をかぶった砂の城のごとくズブズブと崩れ落ちた。けっしていうまいと思っていた「おおぉ」も思わず口から洩れてしまった。

「なかなかやるな」という感想が脳裏から完全に消え去る前に，ゲームをスタート。目の前に広がる多重スクロールの雲々。そこへ間を入れず登場する巨大母艦。自機がカタパルトから放たれる……。なんとも泣かせるドラマチックな演出じゃあーりませんか（ロマンチストの僕は夕陽には弱いのれーす）。そして，高速で後ろから現れるのは，「敵か!?」。いいえ，ミサイルアイテムを置いていってくれました，つまり味方機ね。ゲームが開始されてからほんの数秒の時間の間にプレイヤーをファランクスの世界へトリップさせるのにはもう必要十分の演出。やるな，ズームめ。「ラグーン」のことは許してあげちゃう，うっふーん。

## ファランクスは自機がいい◆◆◆◆◆◆

最近の横スクタイプの自機はどうもカッコ悪いのが多い。「サンダーフォースII」しかり「ナイアス」しかり……。いまどき流線型なんて流行らないんだよね。自機っていうのは，いうならばそのゲームの世界の主人公。ゲームの世界観と完全に融合しうるものでなきゃならないはず。迫りくる敵がゲロゲロのバイオ生物なのに主人公がドラゴン（おっとこれは……？）とかは，やっぱり浮いちゃってるんだよね。よい例ではアイレムの「R-TYPE」シリーズ。独特の文明背景を感じさせるデザインとそれに実にマッチしている敵キャラと背景，お見事。ウルフの「ソルフィース」もなかなか個性的でよかった。タイトーの「ガンフロンティア」もいいな，ちょっと変わりすぎって感じもするけど。

で，「ファランクス」はどうかというとこれが結構独創的なデザインでかっこいい。敵のメカや戦艦とかも，いかにも同一の世界の住人といった感じが出てるし，なんといっても配色がとても系統だっていて，背景ともどもとても世界観溢れるものになっている。

「ラグーン」はともかく，「ジェノサイド」のキャラクターたちも独創的だったし，ズームのデザイナーってかなりのツワモノ揃いみたいだね。背景なんかもほとんど芸術の域に達してる。ズームの社長さーん，いまいるデザイナーさんたちが逃げないよう

X68000用　5"2HD版3枚組　8,800円(税別)
ズーム　☎011(613)0191

Hアイテムがあるなら下で連射

上の赤いところが隠し面への入り口

に椅子に縛りつけておいたほうがいいっすよぉ！

### 面攻略ざんす,ぶらぼー◆◆◆◆◆◆◆

**1面**　敵戦艦の画面奥からの攻撃のときは，アイテムHやLを取っているならば画面左最下部で撃っていればOK。本ボスはアイテムEの波動砲が楽。Lもパワーアップしていれば楽

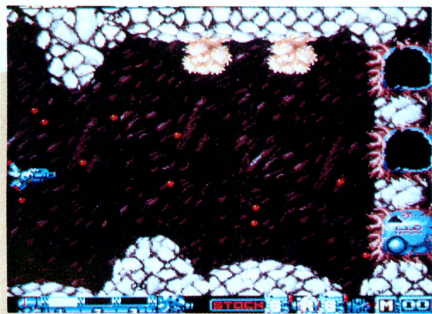
**2面**　写真にもあるような地点が隠し面への入り口。もちろんほかの面にもあるから探してみよう

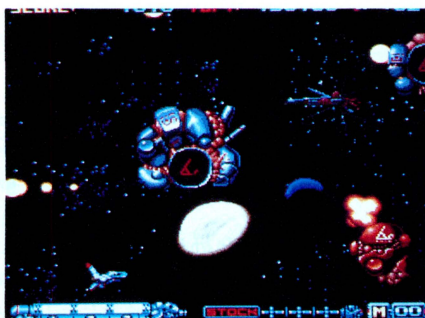
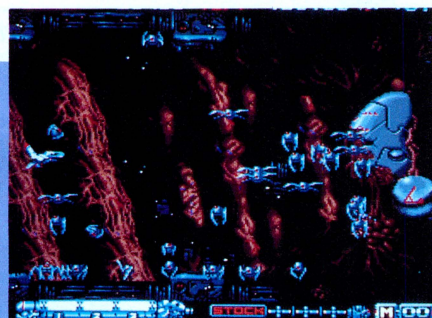
**3面**　初めのほうにある隠し面はパワーアップしていないと地獄。本ボスはアイテムEのブラックホールで1秒勝利。ノーマル装備のときはひたすら撃ちつづける。近づいてきたら画面右上へ避けてそちらへおびきだし，ある程度左があいたら高速で戻って，またショット。誘導弾は早めに察知して避けること

**4面**　画面上部の背景には当たり判定なし

**5面**　1匹目の中ボスは試験管，これをひたすら撃つ。パワーアップしていると楽勝だがノーマルのときはノーダメージクリアはほとんど不可能。私ははじめハマリかと思って焦った

**6面**　面後半の巨大な球形の敵はその直前にあるアイテムEを取ってブラックホールを起こせば楽勝。本ボスの手前の岩は撃たなくて平気

**7面**　上下に揺れるレーザーを撃つザコは先撃ちせよ。慣れてくれば最初のワープゲートまでノーミスでいける。本ボスは上下

3つの分かれ道

広大な宇宙。キレイな背景だ

魔のワープ面

先に上下の砲台を倒せ

にある球状のものを先にやっつけておこう。コウモリガニ（？）を吐き出してきたときには，オプションがある場合は画面最左部に行って打ちつづけていれば結構大丈夫。ノーマル時は反射神経&残機勝負って感じ

**8面**　ネコモドキの隠し面は行ってもあまり得しないかな……。面中盤の中ボスの吐き出す2つのビットは，それぞれを直線で結んだラインの中心が砲撃の対象になることを覚えておこう

### その他,気づいた点◆◆◆◆◆◆◆◆◆

まず，うれしいのがハードディスクへのインストールが可能という点。シューティングゲームとしては日本で初の試みなのかな？　とにかくうれしい。

冒頭でも話した「拡大縮小回転」「ラスタスクロール」などの特殊処理だけど「ファランクス」の場合はさり気なく，しかも効果的に使われていて，なんかとても気分がいい。風呂上がりの牛乳みたいな感じ。1面の多重夕焼け雲に多重工場地帯，2面の水，5面の熱そうな流れるマグマとか，それに画面奥から迫ってくる敵の表現とか，思わず「おおぉ」だよね。プログラマの心意気がガンと伝わってくるよ。

さて，いいことずくめの「ファランクス」だけど，私をはじめ数人のOh!XスタッフからいまいちBGMが面白くないという意見が……(効果音は文句なしだけど)。ノリがなんか「火曜サスペンスドラマ」とかみたい……。

エンディングのスタッフクレジットを見たが，鈴木英樹氏（「ジェノサイド」の作曲者）の名前がないではないか。「ラグーン」の曲も彼ではなかったようだが，あのときは「きっと開発中のシューティングゲーム（つまりファランクス）のほうの曲を担当しているんだろう」と思っていた。あの曲調，個性的な音色，展開，リズム，好きだったのに（ゲーム画面にもろはまっていたよね)。次回作は「ジェノサイド2」だそうだけど……。スズキヒデキー，カンバーック!

あと，せっかくのスピーディなゲーム展開をブチ壊しているシーンがあったので挙げておこう。まず，3面のコース選択シーンと7面のワープシーン。あれはいかんねえ。せっかく壮快にやってるのに，あれはホント気分ブチ壊しって感じ。温泉につかっていてふと見たら毛が浮いていた，みたいな感じよ。それに永久パターンに入れる，つまりわざとぐるぐる回って無限に得点が稼げるからね。

巨大戦艦シーンも，「サンダークロス」や「R-TYPE」のように巨大戦艦側が攻めてきたほうがいい。壊すところを探すのもうんざりだが，それ以上にいちいち画面がブラックアウトしちゃうのがマズイね。理由は上と同じ。

ま，とりあえず気づいた点を挙げてみたけど，3面，5面，7面に関しては意見の分かれるところかもしれないので，ぜひ読者の意見も聞きたいな。

### リアルな世界を徹底的に

あの「アニメ顔」のオープニングと，面と面の間の「アニメ顔」のアイキャッチと「アニメ顔」のエンディングはいらないな。せっかく絶品のリアルな背景グラフィックと敵キャラが「ファランクス」の世界を浮き彫りにしているのに，あの「アニメ顔」を見たとたん「きゅ〜，へなへなへな」って感じ。「ラグーン」とかのARPGの場合のように，人間が前面に出ているような場合はいいとも思うけど。もし「R-TYPE」のエンディングでおめめキラキラの男（あるいは女）が手を振って笑ってたりしたら，私は激怒して「びびんばぁ」とか叫んで，テーブルをひっくり返してたかもしんない。

**総合評価**（0　5　10）

| 項目 | 評価 |
|---|---|
| ゲーム性 | ★★★★★★★★ |
| 熱中度 | ★★★★★★★★★ |
| プログラム技術 | ★★★★★★★★★★ |
| グラフィック | ★★★★★★★★★★ |
| ネコIPL | ★★★★★★★★★★ |
| 音楽 | ★★★★ |

# サソリの尻尾を身につけろ

「ゲーメスト」という雑誌を発行している新声社が，総力を結集して発売したシューティングゲーム「スコルピウス」。サソリの毒は怖いけれど，それを味方につけたとしたらこれほど頼もしいものはないだろう。

Kageyama Hiroaki
影山 裕昭

「ゲーメスト」という雑誌を知っているだろうか？　なに，知らない。本屋にOh!Xなんかと一緒に並んでいると思うが，この本はつまりゲームセンターに置かれているゲームの攻略記事やら情報を提供する雑誌なんだ。ゲーム通の間ではかなり有名な雑誌だろう。その「ゲーメスト」によって作られたゲームが，このスコルピウス。「ゲーメスト」の人たちが作ったゲーム，というだけで僕に遊べるものではないと感じていたのだが，怖いもの見たさという心理が働いたのか，つい手を出してしまった。こうなったらダメもとで「ゲーメスト」に挑戦だあ！

## 3匹のサソリ◆◆◆◆◆◆◆◆◆◆◆◆◆

スコルピウスは，アイテムによって自機がパワーアップしていくタイプのシューティングゲームだ。アイテムは，どこからともなく味方機がやってきて画面上に置いていってくれる。パワーアップアイテムは武器そのものを装備するものだが，なかには画面上の敵を全滅させるというものもある。自機はA,B,C，3つのタイプが用意されていて，ゲームスタート前にこの中からひとつを選択する。自機のタイプによって装備できる武器やパワーアップアイテムの出現パターンが変わってくるので，自分の技術にあった選択がゲームを楽に進めるうえで大事になってくる。

装備できる武器は3WAY，6WAY，リングレーザーなど全部で6種類。武器の中では画面写真にもある稲妻のようなサンダーレーザーが最強を誇っている。このサンダーレーザーのアイテム出現パターンから考えると，Aタイプが前半，Bタイプが中盤，Cタイプが後半に有利になるようだ。

## サソリの生態◆◆◆◆◆◆◆◆◆◆◆◆◆

ユニークなのは自機の操作方法。キーボード，ジョイスティックに加えてマウスでも自機を移動することができる。しかし，マウスでの操作は苦しいかな。Aボタンでショット，Bボタンで触手を出すようになっているが，連射スティックじゃなくてもボタンを押している間は自動的に連射してくれる。だから，連射スティックなら連射スイッチをオフにしておいたほうがいい。

で，Bボタンだが，これを押すと触手がビヨーンと自機から出る。触手はいつでも自由に出すことができて，出したい方向にレバーを入れてBボタンを押せばいい。触手を出すと，触手の先から弾が発射され，自機から弾が発射されなくなってしまう点には注意が必要だ。触手は壊れることがないから敵に直接ぶつけて攻撃すると硬い敵もすばやく倒すことができる。触手を出して自機を移動させると，触手は自機の後ろを追いかけるように動いていく。これをうまく使えば，自機をしつこく追し回してくる敵をやっつけるときに有効だ。

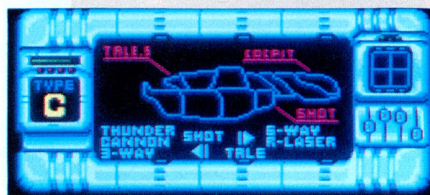
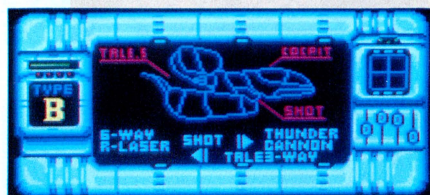
いつまでもぶらぶらと触手を出している姿は，まるで金魚のふん。触手を格納するには再びBボタンを押す。これで再び自機から弾が発射されるようになる。ちなみに，Bボタンを押しっ放しにしていると，触手が伸びたり縮んだりを繰り返す。触手は障害物に阻まれずに伸ばせるし，敵の弾を消すこともできるから，自由自在に操れるようにしておきたい。触手をうまく扱えることがカギになっていることは間違いない。

## 不親切なサソリ，親切なサソリ◆◆◆◆

自機の話が長くなってしまったが，スコルピウスは全部で7ステージ。そのうちステージ3が縦スクロールで，残りは横スクロールだ。背景のグラフィックや，キャラクタデザインなどの色づかいは，さすがに百戦錬磨のソフトハウスが送り出すゲームと比較すると見劣りする点は否めない。

ゲームモードは制作者たちの〝EASYで全面クリアしてもうれしくない〟という考え方から，平(NORMAL)，社長(HARD)しか用意されていない。しかし，そういうことを決めるのは買った人。ちゃんとEASYもつけてほしい。いまのゲームはそれくらいあって当たり前なんだし。

制作者の難易度に対するこだわりは相当なもので，平と社長ではアイテムの出現パターン，敵の配置まで違うものになっている。スタート時の自機の数まで平が5機，社長が10機と違う。コンティニューの回数に限りがないところはうれしいね。

スコルピウスはハードディスクにインス

自分に合った選択を

X68000用　5″2HD版2枚組　7,800円(税別)
新声社　☎03(3293)9321

トールすることもできるようだ。ただし，1Mバイトしか積んでいないマシンではきついものがあるだろう。実際，インストールして遊んでみると，起動時間が速くていい。待たされないということは重要なことだ。エディタでプログラムを書いていて，飽きたら子プロセスでスコルピウスを遊んで，目が疲れたらタイトル画面でブレークキーを押してエディタに戻れば，ちゃんとプログラムが残っている。こんなこともできるのだ。ちなみに我が家のX68000は6Mバイト実装している。

## サソリの活動場所◆◆◆◆◆◆◆◆◆◆◆

それではぼちぼちステージ紹介といこう。ステージ1は惑星の地表が舞台となっている。見た目がギラギラしたステージだ。最初は雑魚キャラのオンパレードであるが，中盤から現れる中型戦艦がカタイのだ。カタイ敵には触手を使おう。触手を戦艦にめり込ませてやれば，素早くかたづけることができるのだ。このステージで触手を思った方向に出せるようになるまで練習しておきたい。ボスはひょろひょろとした足取りでこちらに近づいては，蹴りをかましたりもするが，サンダーレーザーを取っていれば，2秒で死んでしまう。

ステージ2は惑星の地下にある，液体に満たされた生物研究所が舞台。自機は研究所内にあるパイプの中を通らないと先に進めないようになっている。そのため自機の移動範囲が制限されてしまって，けっこう窮屈なステージだ。パイプの中を後ろから敵が迫ってきたりして，「汚ねー」と思ったりもする。触手を後ろへ出してやっつけよう。後半は巨大な敵が前から後ろから迫ってきて押し潰そうとする。こいつも触手をめりこましてやっつけよう。ボスは巨大なハナクソみたいな物体（この表現のほうが「汚ねー」)。パイプの中を動き回って四方から攻撃を仕掛けてくるが，茶色の物体は雑魚で，本体は緑色をしたやつだ。もし，画面上に十字マークのアイテムがあったら，本体が現れるまで取らずにとっておこう。このアイテムは画面上の敵を全滅させる効果があって，ステージ2のボスもアイテムの効果を受けてあっさり死んでしまうからだ。こいつは楽勝だぜ。

縦スクロール面も触手をうまく使え

アイテムを取って抜けろ！

ステージ3は唯一の縦スクロール面。そういえば，あの「ナイアス」にも似たような縦スクロール面があった。3WAYを取っていなければ上へは攻撃できないので，ここでも触手をうまく使おう。ただしサンダーレーザーを装備していれば，このステージもやりやすいはずだ。どうも遊んでいると，サンダーレーザーとほかの武器との破壊力の差が大きすぎるような気がする。

ここでは中型クラスの敵キャラが多く登場する。こいつらも例に洩れず，カタイ。さて，触手を自機の進行方向に出す場合は簡単なんだが，たとえば自機の後ろから迫ってくる敵に対して触手を出そうと思ったら，レバーを一瞬進行方向と逆に入れなければいけない。レバーを逆に倒している時間を短くして一瞬のタイミングで触手を出そうとするが，下手をすると触手が進行方向に出てしまうことがある。だからといってレバーを倒している時間を長くすると，敵に向かって突っ込んでしまう。このステージでハマッてしまったらBタイプを選択してゲームを始めるといいかもしんない。

難しいのはボスも同じ。触手をボスの弱点部分である中心のコアに放り込んでやり，ボスが移動したら触手をコアから外さないように，自機を移動させていく。このあたりの細かい操作が難しいのだが，わりと楽しい。うっかりボスが発射する飛行機を撃ち落としてしまうと，怒り狂ったようにレーザーをすきまなく連射してくる。これを避けるのは大変だから，これは撃ち落とさないようにしたほうがいいだろう。

## 僕のサソリは弱いサソリ◆◆◆◆◆◆◆

僕はステージ3で結構手をやいてしまい，実力の限界を感じつつも，なんとかステージ4にたどりついた。ここは壊された鉱石の採掘現場だ。このステージは背景にオレンジ系の色が使われているので，敵の弾と背景の識別が難しい。さらに敵の中には攻撃すると爆風をまき散らしながら墜落していくのもいる。この爆風に当たったらアウト。だから攻撃しないで避けていくのが正解だろう。このステージでは自機を追いかけてくる虫の格好をした敵がいる。しつこく追いかけ回されたら，触手を出して攻撃すると，けっこう楽にやっつけることができる。僕はこのステージのボスまでたどりつくのがやっとの状態。自分ではシューティングゲームは得意だと思っていたのだが，すっかり自信をなくしてしまった。

ということで，全ステージを紹介することができないのは心苦しいが，興味を持った人は残りステージを自分自身の目で見てね，というありがちな逃げ文句をいってレビューは終わるのであった。

強力！ サンダーレーザー

### 美しいサソリになってほしい

「スコルピウス」はゲームを遊ぶ立場のプロたちが作りあげたゲームだ。しかし，プログラミング技術では，ほかのソフトに比べてやや劣るように思う。なんといっても触手がちょんぎれていてかっこ悪い。それからサンダーレーザーも，もうちょっと稲妻らしい迫力がほしい。はっきりいえるのは，全体的にグラフィックデザインが悪いことだ。これらは見た目の問題だが，見た目のカッコよさも，ゲームにとって大事なことだと思う。

ゲーム内容は最初触手がうまく使えなくてつまらなく思ったが，扱い方がわかってくると，それなりに自分の腕に酔いながらプレイしてしまう。そうなると無限コンティニューのおかげもあって，なかなか遊べるゲームだ。

| 総合評価 | 0　5　10 |
| --- | --- |
| 操作性 | ★★★★★★★ |
| グラフィック | ★★★★★ |
| サウンド | ★★★★★★ |
| 熱中度 | ★★★★★★★ |
| やっぱり触手 | ★★★★★★★★★ |

# メトロポリスは我が掌上に

「鉄道を走らせることによって都市を発展させるゲーム」という売り文句を聞いて，あれを思い浮かべた人もいるでしょう。でも，シリーズとしてはこっちのほうが早くからあったし，面白さも少し違ったものですよ。

Urakawa Hiroyuki
浦川 博之

アートディンクには名脇役という形容がピッタリくるような気がする。必ず新しいアイデアを盛り込んだゲームを発表し，業界に独自の位置を築いているからだ。ヒット作が出るたびにサル真似ゲームが山を築くなかにあって，貴重な存在であることは間違いない。気になる存在だと感じている人はとても多いと思う。

そして，そのアートディンクを一気に主役に押し上げる作品が登場した。このA列車で行こうシリーズ最新作,「A列車で行こうIII」，略して「AIII」である。

## 鉄道王となれ

ニューゲームのウィンドウから「ニュータウン構想」を選ぶと，そこはのどかな農村地帯だ。プレイヤーは鉄道会社の経営者としてこの地を開発する。開発するとはいってもそうややこしいものではなく，鉄道網を整備していれば，駅周辺から次第に発展するようになっているのだ。鉄道と都市開発を密接に結びつけているあたりがいかにも日本のゲームらしい。シムシティーは道路中心だった。

鉄道を開業するには線路を敷いて駅を建て，そばに資材置き場を確保する。そして，車両を買ってダイヤを決め，線路の上に配置すればOKなのだが，いきなりそんなことをいわれても，どこから手をつけたらいいのかわからないだろう。

とりあえずはいまある駅のそばから開発を始めてみよう。最初から敷いてある線路と駅を共有する形で新しい路線を作る。多少は民家をつぶすことになるが，そんなに気にする必要はない。駅があれば家はあとから自然に増えてくるものだ。だから行先も畑のど真ん中でよし。

電車の経費は走った道程，運賃は駅同士の直線距離で決まるので，クネクネした路線はなるべくさけたほうがいい。駅にも経費がかかり，駅同士が近すぎると収益が悪くなる。それさえ頭に入っていれば，どんな路線を目指すかはその人の自由。環状線を敷くのもいいし，複々線でいくのもいい。ただいったん駅の周りが発展してしまったらなかなか拡張は難しいので,「とりあえずこれでいこう」という考え方はしないほうがいい。

車両は客車・貨物列車ともにさまざまな種類がある。スピードが速いもの，遅いもの。2両編成と3両編成。駅を通過できるものとできないもの。値段は高くてもランニングコストを押さえられる高速型がベター。僕はAR-3，211系あたりをよく使う。

引き出しのようなメニューをあちこち開け閉めし，やっと準備が整った。すでに支出欄には巨額の経費が計上されている。経費はその日ごとに手持ちの資金から出ていくので，うっかり資金以上の買い物をすると日付が変わったとたんに“GAME OVER”になってしまう。AIIIではこの「うっかり倒産」が多い。ほかにも，6月1日の税金の支払い日に資金を用意するのを忘れる，「税金倒産」なんてのもある。

いざ営業運転を始めると，まあたいてい儲からない。そもそも町の規模からいって，鉄道の需要があまりないのだから当たり前だ。なかったら作りましょうというわけで，電車が走り出すとプレイヤーは人口を増やすほうに回ることになる。

町を大きくするためにはまず駅の周辺に資材を置く。貨物列車で資材を運んでおけばそれを使ってポツポツと家が立つ。自分で何かを作るときにも資材が必要なので，どの駅にも資材搬入ルートは必ず確保しておこう。これでとりあえずは町は大きくなりはじめる。

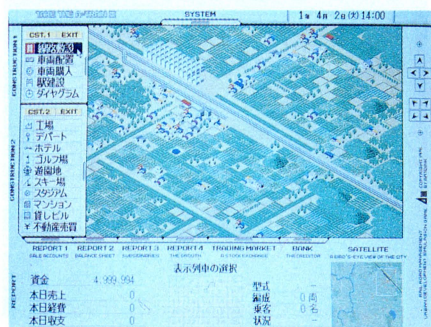
## 都市計画は美学を持て

ただ町ができるのを眺めているだけではAIIIの面白さは半分もわからない。このゲームは自分で計画を立てて町作りを進めていくのが楽しいのだ。

町作りの第1歩は駅ビルタイプの駅舎を建てることから。人口が増えてくると駅の前から道路が伸びはじめ，それに沿って町が発展するようになる。道路の横にある建物は評価額が高くなるという利点もあるので，事前に土地を押さえて建物を建ててしまおう。建てたビルは子会社として経営するので収入も期待できる。

最初はマンション，貸しビルといったあたりがいいだろう。危険も小さく増客効果もある。ホテル，デパートといった商業スペースは，ある程度大きい町の駅前や道路の交差点などに作ると収益が上がる。スタジアム，遊園地といった遊興施設，スキー場，ゴルフ場などのレジャー産業になると曜日や季節によって収益がずいぶん違うので，資金に余裕ができてからのほうがいいだろう。

よほど供給過剰でないかぎり，どんな建物でも建築費用以上の評価額がつく。早い話がビル転がし。ビルでなくスタジアムだ

X68000用　5″2HD版2枚組　9,800円(税込)
ブラザー工業(TAKERU)　☎052(824)2493

どんな町に育てるのか，よく考えよう

ったりすると，1億円単位の壮絶な転がしぶりになる。建てては売り，建てては売りすればあっという間に街並みが完成するわけだ。ただし，転売は年間30件までなので注意。これを忘れると，税金を払うときに資金が調達できず倒産なんてことになる。

こうして町作りに熱中してくると，だんだん自分以外の不動産業者がうっとうしくなってくる。「きさまあ，駅を作ってやったのは誰だと思ってんだ！」といえるようになったら，強欲な資産家の仲間入り。

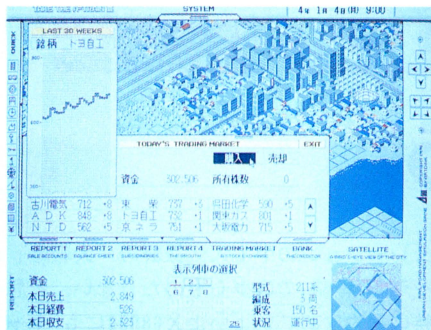
町の風景は昼と夜，そして春夏秋冬にもそれぞれの色づかいがある。刻々と変わる風景を眺めるのも楽しいものだ。夏の土曜日の夜には遊園地に花火が上がったり，クリスマスにはサンタクロースが飛んでいくなんて演出もあって，飽きずに楽しめる。

このAIIIにはゴールがない。レールの総延長が一定数を超えたり，人口が増えると新幹線が開通したり，あるいは資産が一定の額に達すると祝福のメッセージが出たりはする。が，“GAME OVER”以外でプレイが終わることはない。なんらかの理由でプレイヤーが勝手に終わりにするのが相場だ。経営がいったん安定すれば，あとは鉄道と町作りに対する自分の美学だけがプレイを続けさせるパワーである。

## さらなる腕を磨け◆◆◆◆◆◆◆◆◆◆◆◆

アートディンクはセンスがいい。ゲームデザインのセンスもいいし，画面もきれいにまとめている。だが，どうもセンスを優先しすぎるあまり，犠牲になっている部分も少なくない。

まず画面がクォータービューであるために，ビルの背後がどうなっているかわからない。ビルの影でレールのつなぎ替えをするときなど，ほとんど手探りの作業になってしまうのだ。

それから，右ボタンでキャンセルがきかない（なぜか線路敷設のときだけできる）という謎の操作法。なんでわざわざこんなことをするのだろう？　ほかにも「マウス文化の発想」では思いつかないような変な操作法がちょくちょく出てくる。

あとゲームの中では，まず資材置き場。ビルでも建てようと駅前の一等地に土地を確保していると，貨物列車がやってきてそこに資材をドスンと置いていく。資材があると建物が立てられないので，この資材が消費されるのを待つしかない。で，消費されてしまうと資材がないからビルが建てられない。猛烈に腹が立つ。

それから「シミュレーション」のわりに公開されている情報が乏しい。資材はどのくらい遠くの土地まで使えるのか，駅から伸びはじめた道路はどういう条件で伸び，止まるのか，建物の評価額はどうやって決まるのかなど，すべて不明。どうも手探りの部分が多い。

また，「ゲーム内の理屈」が強いのも気になる点だ。たとえば，朝8時にA駅発B駅行きの電車に毎日760名の利用があったとする。複線にして2台同時に発車させるとどうなるか。380名ずつに……，ならない。760名乗った電車が2つできてしまうのである。こういった現実との違いは取っつきにくさにつながる可能性が十分にある。

## A列車の世界に深く落ちよ◆◆◆◆◆◆

でも，やっぱりAIIIは本質的な部分でものすごく面白い。パソコンに向かうとついつい起動してしまう。

シムシティーの真似かという勘繰りはまったく当たらなかった。AIIIはあくまで会社経営のシミュレーションであり，シムシティーとは得られる快楽がまるで違う。

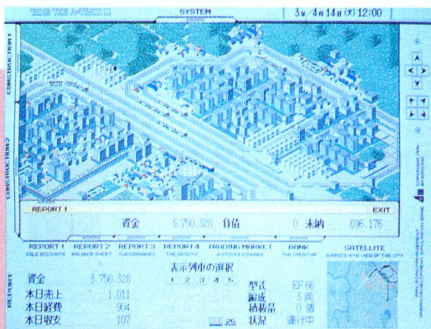
AIIIの楽しみは，まず始めたばかりのときに，こんなふうに路線を敷いて，こういう町を作ってやろうとあれこれ考えることだ。ここまで線路を引っ張ってスキー場を作ろうとか，路線図を思い浮かべながら地図を見て回っているとわくわくする。

資材を使って駅の周辺に建物が建ちはじめたときも，次にどんな建物ができるか，つい目を凝らしてしまう。道路ができたときがいちばんうれしい。町作りが軌道にのった証拠のようなものだからだ。ほかにも初めて黒字が出たときや，ダイヤをいじって収入アップが図れたときなど，「よしよし」と悦にひたれることは多い。

路線を伸ばす。資材を運ぶ。営業を開始する。町が発展する。いろいろ手を加える。発展が一段落すると，また路線を伸ばす。「ここを手直しすればもっと儲かるぞ，もっときれいな町になるぞ」という内なる声につき動かされて延々とこの作業を繰り返してしまう。これがAIIIの麻薬性だ。

まあ，操作性に関しては先にあげた若干の問題点を抱えているので，「A列車で行こうIII ver.2」を期待したい気持ちもなくはないのだが，ともかくAIIIにはAIIIでなければ味わえない楽しさがあるので買って損はない。欧米で発売されるIBM-PC版の評判もいいに違いないと僕は思っている。

財テクで行こう。株式市場

そして，マンション転売

環状線で行こう。端正な街並みだ

### ジャパン・オリジナルの誇りを持って

昨年以来シムシティー，ポピュラスなどのマクロな視点を持つゲームがヒットしている。鉄道を走らせ，都市が形成されていくのを眺めるAIIIもそういったスタイルのソフトだといえるだろう。だが，このAIIIはその流行を追って作られたものではなく，地道にアートディンクのゲームとして磨いてきたのがたまたま（というと失礼か？）流行と一致したにすぎない。だから海外超大作にヒケを取らないほどのクオリティがあり，「AIIIはAIIIだ」といえるだけのオリジナリティもきちんと持っている。これは日本のゲーム界としては実はすごいことなんじゃないだろうか。アートディンクの次回作がさらに楽しみになってきた。

**総合評価**（0　5　10）

| 項目 | 評価 |
|---|---|
| 操作性 | ★★★★★★ |
| ゲームデザイン | ★★★★★★★★★★ |
| ビジュアル | ★★★★★★★★ |
| 音楽 | ★★★★ |
| スピード | ★★★★★★ |
| 熱中度 | ★★★★★★★★★ |

# A-10で行こう

Ogikubo Kei
荻窪　圭

ウォーシミュレーションの中で人気抜群の「大戦略」シリーズ。「スーパー大戦略」では少しものたりない，と思っていた人には特におすすめ。いよいよX68000にも「キャンペーン版大戦略II」の登場です。

リアルタイムに近い映像によって，情報量が増え，入手時間も短縮されたことは確かだが，それだけである。マスメディアが絡むかぎり加工は免れず，現実の一部ではあっても真実ではない。混乱はあっても，正確に摑めるようになるとはいえない。

ここで問われるのが **“想像力”** だ。否応なく流れ込む情報をリアルにするのは想像力であり，その質によって，目にしたものが単なる大戦略になったり，過剰に胸を打ったり，といった違いを呼び起こす。

想像力。SFを読んでさまざまな世界に思いを馳せたり，RPGやSLGに多くのドラマを見出すことのできる我々には，非常に高い経験値がある。質を抜きにすれば。

質は計れない。たとえば，すべてを同一平面上に写像してしまう想像力の質などはどう評価すべきだろう。想像力について考えることなしに，他人が感じたものを非難するのは，村八分精神の露出であって，みっともない。やめてもらいたいものだ。

## 大戦略の歴史

現代大戦略，現代大戦略パワーアップキット，大戦略II，大戦略II/SP，スーパー大戦略，大戦略III，大戦略III'90，キャンペーン版大戦略II。こんなところか。ほとんどPC-9801版だが，ものにより他機種用もある。それぞれがその機種用にカスタマイズされており，移植した時期で登場する兵器が違ったりする。いったいいくつの大戦略が作られたのかは，想像だにできない。

こうなってくると，どの大戦略がいいかという問題が生じてくる。システムソフトのとある人は，現代大戦略パワーアップキットがいちばん，といっているらしい。私個人としては大戦略II/SPだ。

いま，システムソフトが扱っているのはスーパー大戦略，キャンペーン版大戦略II，大戦略III'90の3つ。X68000版にかぎれば，スーパー大戦略と今回発売されたキャンペーン版大戦略IIである。大戦略III'90も出てくるらしい。

とりあえず，うれしいといっておこう。大戦略IIを知るものにとって，スーパー大戦略は「甘口」すぎた。それでもって，大戦略IIIは複雑すぎて，遅すぎて，“ゲームになってない”のだ。

キャンペーン版大戦略IIは大戦略IIと同じではないが，大戦略IIのいいところを引き継いでいて，移植の出来いかんによってはお勧めできる。

X68000用　5"2HD版2枚組　9,800円(税別)
システムソフト　☎092(752)5278

## スーパーとの違い

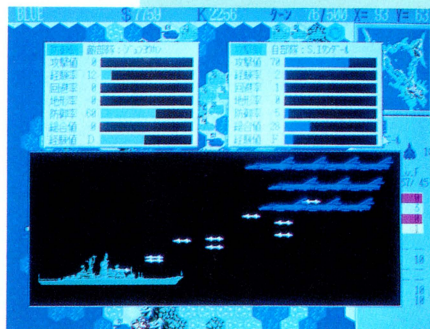
スーパー大戦略は，大戦略IIをベースに，複雑な仕様を外し，甘く，初心者向けにしたものである。キャンペーン版大戦略IIと比べるとその甘口さ加減がすぐわかる。

町の種類。スーパー大戦略では都市と飛行場と首都だけだったが，大戦略IIでは工場と港が加わった。生産に必要なのは金だけだったが，大戦略IIでは工業力も加味される。さらに，耐久度ってえものが加わった。兵士は町の耐久度を0にしないと占領できず，工作隊が町の耐久度を上げると，占領されにくくなり，都市なら資本力が，工場なら工業力がアップするのだ。ノーマルな状態なら耐久度100。これを占領するには，2～3ターン必要だ。

兵器ユニット。大戦略IIには長距離兵器が加わった。6ヘックス先の敵を攻撃できる対空砲や，4ヘックス先を撃てる対地砲。そして，巡洋艦や空母のミサイル。ついでに，町の耐久度を下げて，廃墟と化してしまったり，橋を爆撃して通行不能にできる爆撃機もある。PC-9801版の大戦略IIには隠しコマンドで爆撃機に核兵器を搭載できたが，X68000版ではどうだろうか。廃墟を作れる爆撃機もあるなら，廃墟を立て直す部隊もあって，これは工作車だ。

しかも，だ。キャンペーン版大戦略IIで新たに導入された（何に比べて“新た”なのかは重要である。この場合はスーパー大戦略68Kと大戦略IIだ）ルールがある。代表的なのが **“索敵モード”** の導入。これは“中級”でプレイすべき。間違っても初級にしてはならない。面白みが半減する。

キャンペーン版と銘打ってはいるが，私としては時間を食うキャンペーンモードなどどうでもよい。ただでさえ面白かった大戦略IIのバージョンアップ版，いわば“大戦略II ver.2.0”と捉えるのが正しいのである。正しいのだ。

兵器生産は状況をにらんで

まず艦船を全力で叩け

さらに，スーパー大戦略68Kとキャンペーン版大戦略II（X68000用）にかぎったときの違いを見てみよう。

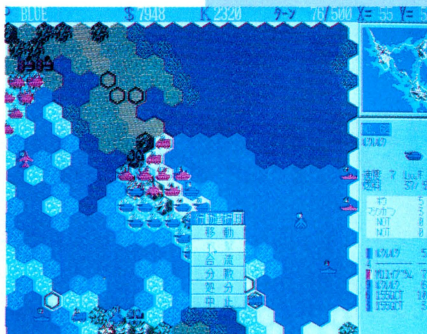
スーパー大戦略68Kは512×512ドットモードであった。また，マルチウィンドウで，ウィンドウの位置や開閉，重ね合わせが自由であった。今回のキャンペーン版大戦略IIは違う。まず，念願の768×512ドットモードなのだ。そして，マルチウィンドウなどではない。その分表示が速くなったのは○だろう。が，画面センスやノリは確実にスーパー大戦略68Kのほうが上だった。そのうえ，画面書き換えがカメだ。とっても残念。つまり，キャンペーン版大戦略のほうが見た目がしょぼい。

## 悩むX68000,待つ人間◆◆◆◆◆◆◆◆

大戦略はいろいろと厄介なゲームである。もっとも厄介なのが，パクパクと時間を食べてしまうことである。考え込むX68000を待つ時間が長い。これはX68000が遅いという問題ではなく，思考ルーチンの問題だ。

キャンペーン版大戦略IIの場合，特に長く感じるのだ。理由は索敵モードである。また出てきた索敵モード。

スーパー大戦略では（大戦略IIもそうだったが），敵の動きがまる見えだった。ボードゲームをもとにしているため，それが当たり前だったのだ。しかし，実際の戦争ではそうはいかないのは当の然。実戦っぽくするためにはどうするか。見えないはずの敵の動きをマスクしてしまえばいい。それが索敵モードの基本概念だ。

だから，索敵モードを中級にすると，敵が何をしているかは，原則として見えなくなる。と，いうことは，だ。敵のターンの間，相手が何をしているかがまったくわからないのだ。敵が自分のユニットに攻撃するとき以外は，ただボォっと全体マップを眺めているしかないのである。ほうら，退屈そうでしょ。長く感じそうでしょ。

よし，一気に……（索敵モード中級）

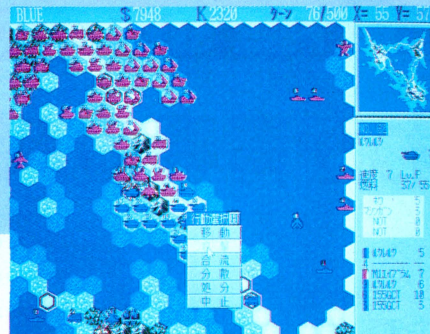
と思ったら，こんなに敵が（索敵モード初級）

X68000が計算しているときに表示されるのは，いま，敵が何の処理をしているかだけである。武装交換，間接攻撃，補給，移動補給，占領，工事，乗車，占領輸送，索敵，合流，攻撃1，攻撃2，移動1，移動2。これだけをすべてのユニットに対して行うのだ。これだけの処理をするとやはり待たされる。何時間ってことはないけど，何十分ってことはある。覚悟をおし。

## なぜ,やめられない◆◆◆◆◆◆◆◆◆◆

つまるところ，このゲームは非常に私の時間を奪う。とても奪う。私だけではなく，X68000の時間をも奪う。コンピュータ側のターンのとき，無為に時が過ぎていく。

なのに，なぜやめられないか。どうして，「あと1ターンだけ」という子供のわがままを繰り返すのか。

大戦略の目的は敵首都の破壊/占領,敵部隊の壊滅である（いいかげん勝利条件に敵の降伏を入れてほしい）。しかし，最終的勝利などどうでもいい。楽しいのは「頭の中で描く小さな作戦の計画とその達成」だ。自ら作り出した物語をディスプレイ上に顕現させること。「イージス艦を沈めてやる」だったり，「うっとうしいM1エイブラムスめ！」だったりする。もちろん，どれも1ターンで終わるわけではない。

だいたい目標をひとつ達成するころには，次のまた次のシナリオができており，目の前の目標があるかぎり，際限なくショートストーリーは湧き出てきて，私はイラク軍を追いかけるアメリカ軍と化してしまう。

しかしなあ，どうもアメリカ軍を叩くことばかりに気がいっていけねえや。ま，強きをくじくのは男の基本だあね。

## 疑似体験と想像力◆◆◆◆◆◆◆◆◆◆◆

ミサイルからの映像を称して「テレビゲーム」みたいだといった人が，平和を語る人々から非難されていたが，どっちもどっち。現実の映像には違いないが，我々の網膜に届くまでの間にはいくつものフィルタがかかっているため，そこから何かを感じるためには想像力を必要とされるからだ。

「ゲーム」だと感じるためにはそういうゲームで遊ぶ経験，ないしはイメージが必要である。テレビゲームを忌み嫌っている人からみれば，あの映像を「ゲーム」だというのは不気味以外の何物でもない。

もし，戦争を見て「テレビゲームみたいだ」と思ってしまう自分を許せないなら，大戦略に手を出してはいけない。アエラのおじさんがボードに描いた地図を見て，あのエイブラムスをこっちに移動して，この部隊は囮にして，それで4ターンかな，と考えている自分を発見するからだ。

メディアによる疑似体験の積み重ねは，現体験と疑似体験の差異を縮め，ついにはより多く詰め込まれた疑似体験の感覚がその人の想像力を決定してしまう。冒頭で述べた，どんなものも同一平面上に写像してしまうような類の想像力が出来上がる。単純で現実離れしたお話でも，毎日毎週，繰り返し接していれば，それがどんな影響を及ぼすかわかったものではない。資本主義的文明ってのはそういうものなのだ。

### 大戦略を知らない人々へ捧げる総評

大戦略というのは，歴史のある現代戦シミュレーションだ。架空のマップの上で，実在の軍をモデルにして戦争をする。伝統的なボードゲームのルールが基本になっているので，1回の戦闘はターンという単位で行われ，参加している軍が順番に自分たちの部隊を操作する。ひとつの部隊は1種類のユニットで構成され，ひとつのヘックスにはひとつの部隊しか置けない。ある部隊が敵の部隊と隣接するヘックスに到着すると，戦闘が始まる。

今回のキャンペーン版大戦略IIや，もうすぐ発売されるであろう大戦略III'90になると，ボードゲームの呪縛から逃れようと，新しい試みが導入されている。

あらかじめシナリオの決まった過去の出来事をシミュレートする光栄のものやアートディンクの戦争シミュレーションと異なり，大戦略は架空のマップ/現実の兵器で戦えるという点で秀逸である。一度は遊んでみる価値はある。問題はどの大戦略にするかという点だけだ。

もし，大戦略に割くだけの時間があれば，キャンペーン版大戦略IIを購入してみるべきだろう。そうすれば，君も次の戦争では立派な評論家である。興味はなくとも，やっていれば兵器の名前や特徴は覚えてしまうものだ。戦争報道を見ると，知っている名前ばかり出てきて驚くことになる。スカッドミサイルやパトリオットは出てこないけどね。

| 総合評価 | 0　　　　5　　　　10 |
|---|---|
| 操作性 | ★★★★★★★ |
| 箱庭度 | ★★★★★★ |
| 軍事おたく度 | ★★★★★★★★ |
| サウンドセンス | ★★★★ |
| 戦略度 | ★★★★★★★★（大） |

# わが青春のパロディウスだ!

Nakamori Akira
中森 章

こんなにいいゲームを1回だけのゲームレビューで終わらせるのはもったいない。今月は「パロディウスだ！」にとりつかれてしまった中森さんが，思い入れたっぷりに，このゲームの魅力を語ってくれます。

## 突然の出会いだ,パロディウスだ!◆◆◆

「踊り子のねーちゃんがおるんや」。友人が突然しゃべりだした。「タコがおってな，ねーちゃんの股をくぐるんや。すると，ねーちゃんが腰を振るんや」。

あの日，友人がふと口にしたゲームが「パロディウスだ！」だった。シューティングゲームの苦手な僕は，それまでゲームセンターでシューティングゲームをプレイすることはなかった。下手なところを人に見られるのが恥ずかしいという思いもあったが，熱中できるゲームがなかったのが大きな理由だ。もっぱら，脱ぎマージャンやテトリスに闘志を燃やしていた。それが，友人の言葉をきっかけとして，事情が変わってしまった。

タコが空を飛んでいて，ねーちゃんの股をくぐるゲーム。この怪しげなゲームが僕の頭の中にこびりついて離れなくなり，早速ゲームセンターに出かけてプレイすることにした。はたして，「パロディウスだ！」はどのゲームセンターでもすぐに見つけることができた。その頃，「パロディウスだ！」は人気の最盛期で，そのテーブルの前に人が座っていないときがないくらいのモテモテぶりだったのだ。

さて，実際に「パロディウスだ！」をプレイしてみた感想は「かわいい」のひと言だ。やっぱりタコがいい。ぽっぽっとお尻から煙（？）を出しながら宇宙を進む雄姿。飛び散る汗。そしてやられたときに飛び出す目玉。こんな健気なタコを見て，いとおしく思わないわけがない。僕は暇があるごとにゲームセンターへ出かけては「パロディウスだ！」をプレイするようになっていった。そして，このゲームが早くX68000に移植されないかなという思いがどんどん募っていったのだ。

## 必死の思いだ,パロディウスだ!◆◆◆◆

「パロディウスだ！」がX68000に移植されるという知らせを聞いたのはいつのことだったろう。かなり前のような気がするが，「やったね」と飛び上がって喜んだのを覚えている。当時は早く情報を知りたくて，やきもきしながら毎週のようにコナミのテレホンサービスを聞いていた。ファミコン版の「パロディウスだ！」発売後はX68000版の情報が少なくなってすこし不安になったが，信じるものは救われる。1991年4月19日，やっとのことでX68000版「パロディウスだ！」が発売された。

その日は朝からそわそわして，会う人ごとに「今日は〝パロディウスだ！〟の発売日だよ」と宣伝して回ったものだ。やたらとはしゃぎまわる僕は，友人たちの目にはどんなふうに映ったのだろう。しかし，そんなことより，僕の心配は売り切れだ。

4月19日は休暇を取って買いにいこうかとも思ったが，さすがに仕事が忙しくて休暇を取れる状況ではなく，泣く泣く5時半の終業時刻を待っていた（僕は一応サラリーマンなのだ）。しかし，さすがに残業はやる気もなく，終業のチャイムとともに会社を飛び出して電車に乗り，その30分後には憧れの「パロディウスだ！」のパッケージを手にしていた。そして，次の日が休日(土曜日)であるのをいいことに夜遅くまで「パロディウスだ！」をプレイしつづけたのだった。

ゲームセンターでは富士山の面（嗚呼！日本旅情）で死んでいた僕も無限コンティニューのお陰で最終面を見ることができて大感激だった。時刻は午前4時になろうとしていた。そして，その日は幸福な気持ちで眠りについた。

## まだまだ現役,パロディウスだ!◆◆◆◆

いまでも「パロディウスだ！」はどこのゲームセンターにも置いてある現役バリバリのゲームだ。それがX68000に移植されたのは画期的なことだ。

X68000のゲームにはアーケードゲームからの移植が多いが，そのゲームが発売されたとき，オリジナルのゲームをゲームセンターで遊べることは少ない。移植作の多くは何年か前にゲームセンターで話題になったゲームなのだ。ところが今回は違う。X68000で練習をして，ゲームセンターでいい格好をするということができるのだ。昔はあんなに難しく感じたゲームセンターの「パロディウスだ！」だが，いまではお金さえ注ぎ込めば（コンティニューを続ければ）最終面までは行けそうな気がするから不思議なものだ。

## やっぱりオートだ,パロディウスだ!◆◆

僕はオート（パワーアップ）モードの愛好者である。いまにして思えば，編集部で「パロディウスだ！」をプレイしているときに耳の後ろに感じた生暖かい風は……。

X68000用 5"2HD2枚組 9,800円(税別)
コナミ エンタテイメント ☎03(3264)5678

うーん，星座がきれい

先月のレビューを読んでやっと謎が解けた。西川善司さんのいる前では今後プレイをするのは控えることにしよう。

と，冗談はともかく，オートモードあっての「パロディウスだ！」，AI戦闘モードあっての「ドラクエIV」なのだと，私は思う（「桃太郎伝説II」のオートバトルはいらないと思うが）。オートモードを使っていれば，パワーアップのことを心配することなく，敵の弾を避けることに神経を集中できる。

これこそシューティングゲームが苦手な人への福音だ。ゲームセンターで初めてこのモードを見たとき，頭を100tハンマーで殴られたような気分になった。これぞ人類の革新，コスモバビロニア主義だ（何のこっちゃ）。断言できる。もし，オートモードがなかったら，いくらキャラクターが可愛くても，これほどまでに「パロディウスだ！」にのめり込むことはなかったであろう。

## 気分よくゲームを,パロディウスだ！◆◆

X68000版の「パロディウスだ！」の特徴のひとつが「アーケードのまんまの完全移植」ということであるが，実はちょっと違う。画面の横幅が短いような気がする。僕は2面目（ピエロの涙も3度まで）の終わりでは踊り子のねーちゃんの股の下にずっといるのが好きだった。でも，X68000版では股の下でうろうろしていたら画面の端とねーちゃんの足に挟まれて死んでしまった。ちょっとショックが大きかった。

まあ，これは些細なことなので移植の素晴らしさを損なうものではないということを断っておこう。それよりも気になるのは面の変わり目だ。約5秒間ほど画面の動きが止まってしまう。これは気分よくやっているゲームの流れを損なうのでちょっと改造することにした。と，いってもプロテクト破りではないので勘違いしないように。また，以下の内容がわかる人だけがやるように。中途半端な改造は事故のもと。

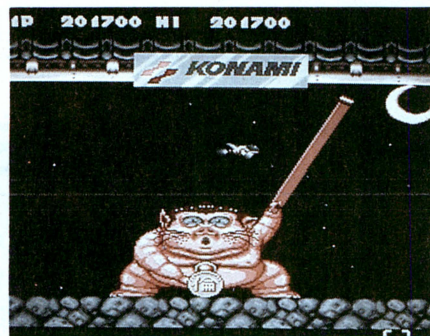
富士は日本一の逆火山

もろだしでツインビーの勝ち

面の変わり目で動きが止まるのはあきらかに，フロッピーディスクへデータを読みにいっているせいである。そこで，アクセスが速い別のメディアから読み込ませることにすれば，画面が止まる時間を短くすることができる。このためにディスクキャッシュを利用するのもひとつの手であるが，僕はRAMディスクを使うことにした。

まず，容量が1.2MバイトのRAMディスクを2つ（ドライブ0とドライブ1用）用意する。そこに「パロディウスだ！」のディスクの内容をすべてコピーし，そこからデータを読み込ませるようにするのだ。ドライブの名前の付け替えはSUBST.Xコマンドを用いる。最初はOh!Xのおまけディスクの真似をしてDRIVE.Xを使おうと思ったが，RAMディスクのルートディレクトリに置けるファイル数が100個程度なのであきらめた。

ディスク2には約140個のファイルがあるので，サブディレクトリを作り，そこにファイルをコピーすることになる。このためドライブの名前を付け替えるDRIVE.XではなくSUBST.X，あるディレクトリをドライブとして参照するSUBST.Xが必要になる。

基本的にはSUBST.XでRAMディスク上のディレクトリをA:,B:という名前に変えたあと，ドライブ0に「パロディウスだ！」のディスク1を入れて本体のPARO.Xを起動すればよい（サウンドドライバは登録しておく必要がある）。このとき，ドライブ1側にも適当なディスクを入れておかなければならないので注意。プログラムはドライブ1にディスクが入っていることを確認してから実際にディスクへアクセスする構造になっているようだ（ファイルはRAMディスクから読まれるのだが）。

この改造により，面の切り替わりでの待ち時間が1秒程度になった。いやあ，楽ちん楽ちん。ただし，要3Mバイトである。RAMディスクでなくハードディスクを利用する方法もある。このときは待ち時間は2秒程度に増えるが，RAMを増設している必要はない。こっちのほうがお勧めの方法かな。

## 最後にひと言,パロディウスだ！◆◆◆◆

ファミコン版を見て少し不安になっていた僕もX68000版の「パロディウスだ！」には大満足だ。本当に，

**見頃。食べ頃。野口五郎一つ！**

な出来なのだ。ノーミスのデモパターンや再現プレイ，ハイスコアの保存など，いくつか要求はあるけれど，本道であるゲーム自体がよくできているのでそんな考えもどこかへ吹き飛んでしまう。明日，宇宙の終わりが来るとしても僕は「パロディウスだ！」をやり続けることだろう。
P.S. パロディウスだ！のサウンドドライバをコマンドモードから起動すると「見つかっちゃった」という声とともに，サウンドテストモード(要はミュージックモード)に入ることを知ってました？

ベルパワーのひとつ，メガホン攻撃

### BGMの不思議

えええと，この音楽は何だっけ。「パロディウスだ！」をプレイしていると，小中学生のとき音楽の時間に聞いたことのある曲が流れてくる。曲名は喉まで出かかっているのに……。

このもどかしさを解消するために，僕は「パロディウスだ！」のサウンドトラックCDを買ってしまった。で，買ってきたCDの説明書を見て不思議に思ったことがある。それは，ゲームの最後に，

We love GRADIUS Ⅰ

と表示されるわりには，クラシック音楽から引用してきたフレーズ以外のBGMが「グラディウスII」の音楽ばかりなのはなぜだろうということだ。だれか教えてちょうだい。

| 総合評価 | 0 — 5 — 10 |
|---|---|
| かわいい | ★★★★★★★★★★ |
| 音楽 | ★★★★★★★★★ |
| 現役度 | ★★★★★★★★★ |
| 日本旅情 | ★★★★★★★★ |
| 熱中度 | ★★★★★★★★★★ |

# それは2億年前の物語

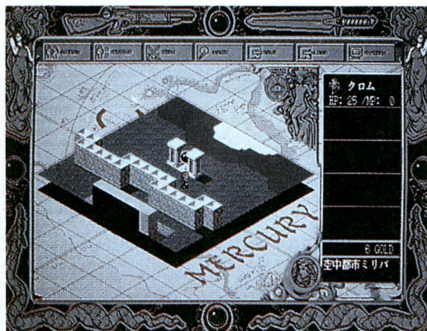
「2億年もの太古の世界に繰り広げられる想像を超えた世界」というふれこみどおり，この「マーキュリー」は2億年前の地球を舞台にしたRPG。ここは魔法と科学が共存する世界，科学文明に育った「クロム」の運命や如何に。

Takahasi Tetsushi
高橋 哲史

さてさて，新進気鋭のソフトハウス，マキシマが放つ大河RPG「マーキュリー」です。PC-9801からの移植ということですが，さてその完成度のほうはどうでしょうね。X68000ユーザーの皆さんは目が肥えていらっしゃるから，ただの移植じゃ満足しないよ，きっと。うん。それではさっそく起動して2億年前の世界にレッツゴー！

## 2億年前って,どれくらい前？ ◆◆◆◆◆

……2億年前の地球。世界は大きく3つに分かれながらも，それぞれの隆盛を誇っていた。魔法，呪術を社会の中心に据えたファンテュラ族とアポークリフ族，そして科学文明を著しく発展させたエンデ族。ファンテュラとアポークリフはそれぞれ異系統の魔法，呪術を崇拝していたので，ことあるごとに互いに反目しあっていた。また，エンデも科学工場などからの汚染物質の排出を長年続けていたので，ほかの2部族からの反感をかっていた。

このように一見栄華を極めながらも，世界は一触即発の緊張状態にあった。世界の均衡を辛くも保っていた8人委員会がその一員であるブランドの策謀によって消滅，地球全土は戦火の渦に巻き込まれていく。この戦いでファンテュラが世界を掌握してから約700年後の物語である。

街外れに不思議な門が

X68000用　5"2HD2枚組　8,800円(税別)
マキシマ　☎06(561)2215

## みんな間抜けなのね ◆◆◆◆◆◆◆◆

さて，主人公クロムはエンデの空中小都市ミリバで量子時空論を専攻する学生であったが，ここにも侵略の手が。母親を殺されたクロムは地上に降り，ファンテュラの帝王を倒すことを決意する。
クロム「母ちゃん，見ててくれよ。俺はやるぜ！」
といいつつも，サラマーの村に降り立った瞬間に捕らわれの身となる，なんとも間抜けなクロムであった。しかし，門番やそのほかの皆さんもかなりお間抜けなので，すぐに脱獄。その後，ジラトの宮殿でランカーを倒したり，魔都市フラルゴンで“バンザイおじさん”の娘を助けたりしながら冒険を続ける。

まあ，こういう具合にストーリーは進んでいきます。システムのほうでは，戦闘時には魔法や光線を使うと視覚的に効果を確認できたりするのでなかなか楽しめます。

また，モンスターのネーミングセンスがなかなかすごくて，結構笑っちゃうのが多いというのも特徴でしょう。先のバンザイおじさん，月面人，パンサーウーマンとかはまだいいほうで，悪のネズミ（あう）とかハンドパワー（あうあう）とか，もうたまんないっす（笑）。

## うまく消化してね ◆◆◆◆◆◆◆◆◆◆

全体的な感想としてはよくまとまっていますが，まだ詰めが甘いな～という感じです。BGM，グラフィック，システムなど無難にこなしているのですが，いまひとつ新鮮味に欠け，プレイしていてマンネリに陥ってしまう時期が早いと思われます。

“アイソメトリックビュー”も奇抜なだけであまり効果をあげていませんしね。移動の際マイキャラが画面の端まで行かないとマップがスクロールしないため，見通しが悪かったりと，かえって逆効果になっています。また，なんとなく「ついでにX68000も」といった雰囲気が拭いきれません。フルマウスオペレーションなのにキーボードのほうがやりやすくできているし。

見合って，見合ってえ～

ほかにディスクを読みにいく際，本当に黄泉にいってしまって還ってこないことがあります。このおかげで私は3度ほどキャラクターを消失しましたので，やや怒りを込めてここに記しておきます（私のドライブが悪いせいではないと思います）。

しかし，グラフィックなどはきちんと直しが入っており，「X68000！」といった画面になっているので，その点の努力はGoodですね（ただ，あまり多色に慣れていない人が描いたような気がするけど）。気軽にプレイする分にはなかなかいいのではないかとは思いました。

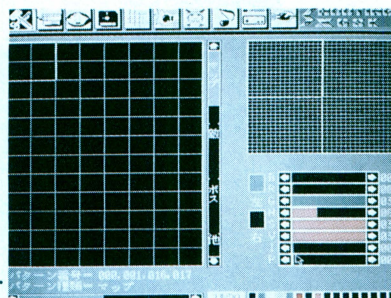
### 良い子のためのRPG

本文中でかなり厳しく書いてしまいましたが，この「マーキュリー」はやっている間は決して面白くないわけではありません。数々のアイテム，魔法，ダンジョン，モンスターに自動戦闘と，RPGに必要な要素はきっちりと盛り込まれていますし，謎や仲間集めもそれなりに演出されています。いうなれば「RPGってどんなのだろ？　私，初めてだからどきどきしちゃう」といったようなまだすれていない良い子のためのRPGといえると思います。

| 総合評価 | 0　5　10 |
|---|---|
| グラフィック | ★★★★★★ |
| サウンド | ★★★★★★★ |
| シナリオ | ★★★★★★ |
| 操作性（キーボード） | ★★★★★★★★ |
| 操作性（マウス） | ★★★★ |

# 簡単操作でゲーム作り

Yokouchi Takeshi
横内 威至

プログラム不要。マウス一丁で手軽にシューティングゲームが作れるコンストラクションツールです。レビューはアマチュアゲームプログラマとしてはトップレベルのあの人。X1turboグラディウスの横内君にお願いしました。

どうも皆さん，横内です。いろいろな事情で北海道，札幌にて書いています。やはりここは寒いです。で，そんなときは部屋でX68000(借り物)ででも遊ぶのがなによ り。とはいっても近頃のゲームは軟弱だしもう飽きた，なんていうときにピッタリなのはこの1本,「シューティング68K」でしょう。

## なんのソフトだ?◆◆◆◆◆◆◆◆◆◆

単純明快，これはシューティングゲームを作るソフトです，といってもピンとこないでしょう。要するにこれは，ゲーム作りで面倒なプログラムに手を染めずに，面白い部分だけ考えてゲームを作ろうというツールなのです。そうです，あなたはグラフィックを描き，背景を組み立て，敵の動きを考えたりするだけでゲームが作れるってわけなのです。

もしかしたら皆さんもあるんじゃあないでしょうか。ゲームを作ってみようとしてグラフィックだけは作ってみた，なんてことが。そんなものを引っぱり出して利用するチャンスでしょう。

このソフトは誰にでも簡単に遊べるように作られているため，ツール仕立てのソフトになっています。すべてマウス操作によるアイコン選択で行えます。そのため，テストプレイと新規ファイル名入力以外，すべてマウス操作で手軽にできます。

しかし，このようなツールゆえにいろいろと制限されることもあります。まずいい忘れましたが，このツールは縦スクロールのシューティングゲーム専用です。そしてパワーアップなどはお決まりのものからしか選択できません。

あなたが作るのはキャラクター，背景，敵の動き，ボスの動き，その他ちょっとしたこととなっています。細かいことは，とりあえずサンプルを作ってみながら挙げてみましょう。

## まずはキャラクターを◆◆◆◆◆◆◆◆

最初にメインのキャラクターが必要です。背景用のヤツ，ザコの敵キャラ，ボス，自機，いろいろな弾，パワーアップアイテム，その他，数字などゲーム中に表示されるものを描きます。面倒だったらサンプルプログラムから拾ってくるのもいいでしょう。

スプライトエディタの機能としてはライン，円，反転，コピー，回転ほか，だいたい使いそうなものが揃っています。バリバリ描いていきましょう。

パレットテーブルで色を作り，右上のウィンドウでパターンを作ります。パレットテーブルの左端は透明色，右端は点滅色です。色は何度も変えながらやるのですが，パレットテーブルは最下段にあってウィンドウからけっこう遠くて困ります。せめて真ん中にしてほしかったところです。また，スプライト，パレットのコピーはエディットとは別のアイコンのため，少し面倒ですね。まあいいけど。

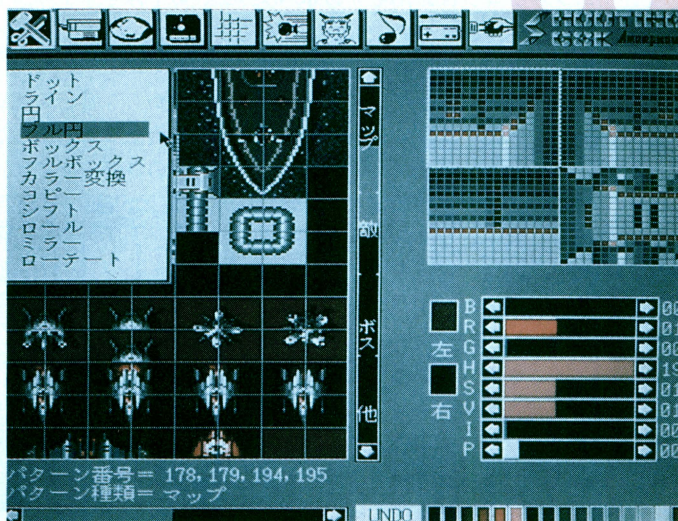
少し描いたら別アイコンのマップエディタで背景を作ってみたり，交互にやるとよいでしょう。

## 勝負する敵の製造◆◆◆◆◆◆◆◆◆

では敵をいろいろ作ってみましょう。全部で64種類の敵が作れます(1ステージあたり)。スピード，動き，固さ，点数，弾の撃ち方，パワーアイテムのことなんかを自由に設定できます。

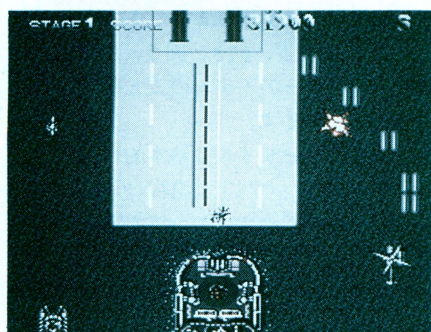
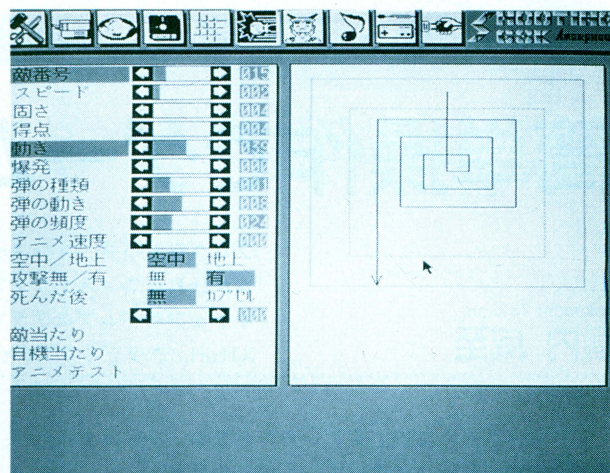
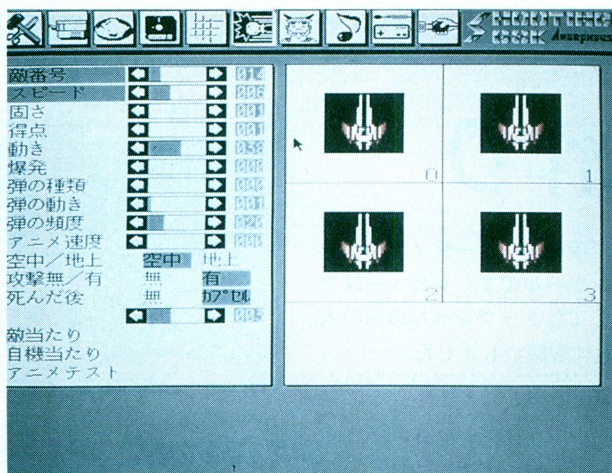
キャラクターサイズは基本的に32×32ドットです。当たり判定は自由に設定できますのでこれより小さいキャラクターなら大丈夫です。ちょっとした中ボスだとかは難しいかもしれません。私は2個のキャラクターを組み合わせてちょっと大きなキャラクターを作ってみました。

動き方は基本的に，画面に現れた位置からどういう軌道を描いていくかを指定するという仕様になっています。これはデフォルトで64種類設定されていますのでそのなかから選ぶか，マウスで手描きパターンを登録します。一応，追っ掛けパターンなどもありますが，自機にあわせた動きができません。X，Y座標があうと突っ込んでくる，だとかのちょっとシャレた動きはできません。また敵を生み出すハッチだとかも難しいでしょう。

X68000用　5"2HD 2枚組　6,800円(税込)
ブラザー工業(TAKERU)　☎052(824)2493



これがスプライトエディタ部。作業の中心となる

敵の設定を行うところ

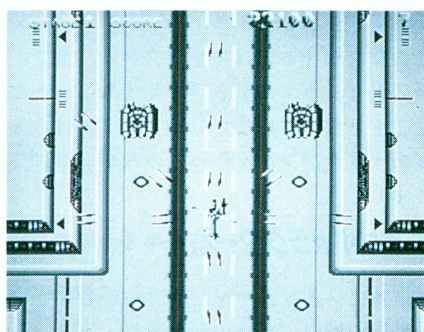
飛行パターンはこんなぐあい

パワーアップの例。レーザーは敵を貫通する

パワーアップの例。最強の6WAYだ

弾もだいたい同様で16パターンの軌道のなかから選んで設定します。なぜか自分のいるとこ目指して撃ってくる，ってヤツがないんです。これはちょっと困った。撃つタイミングも一定だから，なんてゆうか，敵が皆バカに見えてしまいます。なんとかなればもっと面白くできると思うんですけど。そのほか，弾の種類や撃つ頻度，弾の速度なども設定します。

敵の動きが決まればほとんど決まり。あとは細かいことをバンバン決めていくだけです。アニメーションはお決まりの4パターンが一定のタイミングで表示されるだけなんですが，パターンごとに判定が決められるのでコアが開いたときだけダメージを食らう，なんてヤツができます。まあOKでしょう。

パワーアップアイテムの設定もします。それなりの奴にはそれなりのキックバックをもらわないと面白くないものです。レーザー，?-WAY，無敵，1UPから適当に選びます。

おっと，ここでもうひとつ忘れていました。板付き（背景スクロールと同期して動くキャラクター）用に空中物か地上物かの選択もできます。これはキャラクターが重なったときの表示や自機が当たったときに死ぬかどうかでからんできます。しかし，プレイ中には画面中4個までしか地上物キャラが出せないようなので，地上を動く戦車なんかがほしくて，ちょっと困ったものです。バージョンアップかなんかでなんとかしてほしいですね。

ま，なんとか適当に敵も作ってみました。やはりデザインも難しいものです。なんとなくグロ◯ダーっぽいのとか変なのばかりできました。まあ，ご愛敬ということで。今回はこんなもので勘弁してください。

さて，いくつか敵ができたらさらに次にいってみましょう。

## 舞台を飾る◆◆◆◆◆◆◆◆◆◆◆◆◆◆

左から3番目の地球儀みたいなアイコンが舞台セットです。ここではメインスクロールの背景，サブスクロールの背景，および敵出現位置のセットを行います。

メイン，サブ背景は最初に描いておいたグラフィックの背景部分のパネルを組み合わせて作っていきます。機能はフリーサイズのパネルをGETして置いていくのと，背景から背景へのコピーができます。ここはなかなか楽しく作っていけることでしょう。

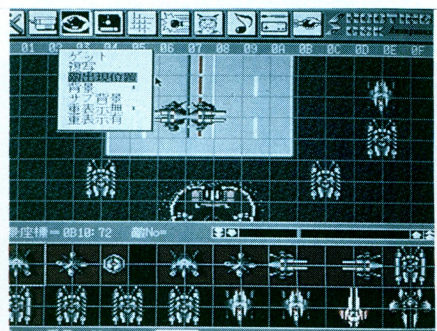
しかしメイン背景とサブ背景が完全に独立しているため，うまく作らないと狙いどおりの2重スクロールにはなりません。テストを重ねてがんばってみてください。またスクロールスピードは音符のアイコンで設定できます。うまく調節しながら作っていきましょう。

ここで敵の出現位置を設定していくこともできます。敵はメインスクロール部分に置いていきます。1カ所につきひとつずつ置けます。これもうまく考えながらがんばってみてください。結局ここですべてゲームバランスが決定することでしょう。バリバリとセンスを発揮してください。

ここが終わればあとわずかで1ステージ完成です。ではいよいよ，ラストを飾るボスキャラの作成です。

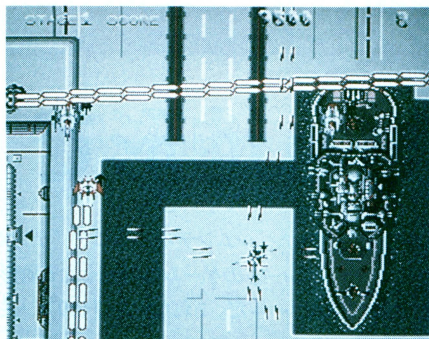
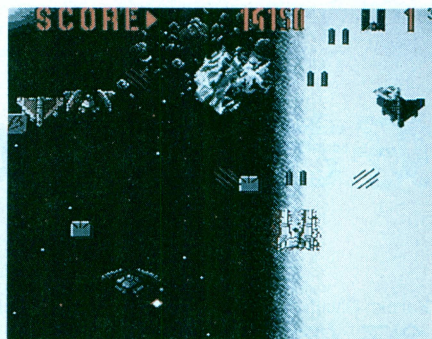
## 男の勝負◆◆◆◆◆◆◆◆◆◆◆◆◆◆

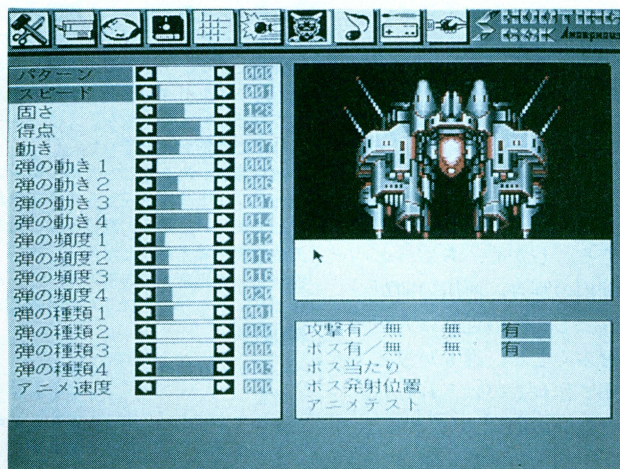
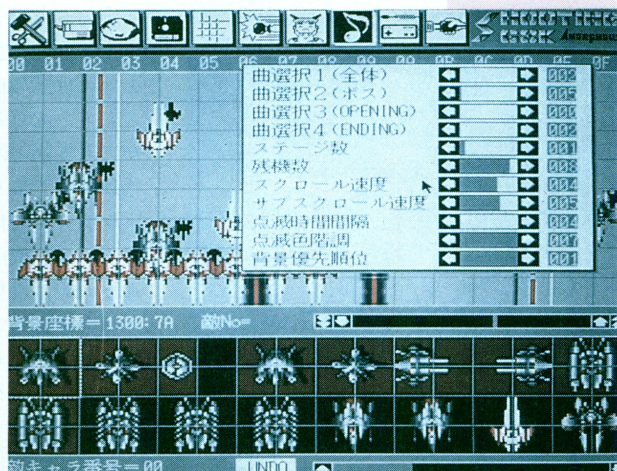
残すヘヴィな作業はあとこれだけです。ステージを締めるボスを作らねばなりませ

マップエディタでは下のパーツを置いていく。簡単

同様に敵キャラの出現位置を決める

ん。
　ボスは128×64ドットで４パターンのアニメーションとなっています。攻撃は最高４カ所から弾を撃てます。それぞれ弾の動き方，形，周期などが設定できます。
　動き方もいろいろ用意してありますが，やはり自機の動きにうまくあわせてくるようなものはありません。まあそれなりにうまく作れるものでしょう。
　固さや点数，スピードなんかもメリハリのつくよう，適当に作ってみましょう。
　ちなみに，ボスを出さないように設定すると，画面中に残っている敵を全滅させれば面クリアとなるようになっています。地上物をうまく使えば最終防衛ラインみたいなものが作れることでしょう。センスにまかせてアバウトに攻めてください。
　やはりこれも僕はパクることしかできませんでした。見て一発，テト○ンらしきものになってしまいました。はぁ，この手のセンスさえあればいいのに。皆さんはパクったりせずがんばってみてください。
　あとは面のBGM，ボスのBGM，さっきのスクロールスピード，点滅色の周期，階調なんかを設定すればついに１ステージの完成です。けっこう簡単に作れるものでしょう。
　あとは数ステージ同じように作るだけ。その他タイトル，エンディングなんかの画面も描いてください。ようやく１作仕上がりです。ごくろうさまでした。

## ツールの未来◆◆◆◆◆◆◆◆◆◆◆◆◆

　どうでしょうか。なかなかに面白いものでしょう。まともにこんなゲームを作ろうとしたら何カ月かかるかわかりません。
　しかしツールゆえにいろいろと制限がついてくるのも事実です。
　細かく見てみるとやはり動き方の制約が大きく出ていることと思います。「誰にでも簡単に」というポリシーを崩すものとなるかもしれませんが，簡単な専用コンパイラなどを載せておくと，かなり使えるものになることでしょう。
　欲をいわせてもらえば，パワーアップなんかももっと凝ってほしかったところです。最近では定版となった多種多用のオプション，その他ありがちなものは入れておいてもよかったかなとは思います。そうですね，縦スクロール究極のゲーム，究○タイガーぐらいのものが作れるようになったらもう文句ありません。RAI○ENなんかでももちろんOKです。しかしここまで完成されたものをプログラミングなしで要求するのは虫がよすぎるというものでしょうか。
　さて，あらゆることが特殊化されて，随所に専用プログラムが必要なゲームばかりになった現在，ユーザー単位で満足なゲームを作るのはたいへん難しくなっています。プロの人たちは専用ツールないしはシステムを使用しているとはいえ，こんな状況では最低限の部分のみで開発を助ける，まさに小道具でしかないことでしょう。
　我々ユーザーに必要なツールとはどんなものでしょうか。我々はクリエイティブにコンピュータを使うべきです。ツールは使って便利，から使って楽しいものに生まれ変わってくるべきしょう。ならば最低限の労力で単純ながらも手軽にアミューズメントとなるこのツールはかなり革新的なものではないでしょうか。
　そう，ひとりで夢を，楽しみを求めるにはこういうかたちのツールがベストでしょう。このソフトはまだ完成された形態にはなっていないかもしれませんが，少なくとも新しい可能性を開いたことは間違いありません。いままでのどんなゲーム，どんなツールにもなかった，新しいクリエイティブな楽しみが得られることは確実です。
　くどいようではありますが，コンピュータをただのゲームマシンで終わらすことは非常に無駄なことと思います。これだけ開放された世界で，いつまでも受け身の体勢でいることはこのパーソナルコンピュータの意義に反しているとは思いませんか。今からこの広がりすぎた世界に手を出すのはたしかに大変なことでしょう。
　そう，だからこんなにも簡単に，クリエイティブに楽しめるソフトを僕はすすめます。パーソナルユースのコンピュータの，最大の楽しみを与えてくれるソフトが出たのですから。
　ということで，いずれまたお会いしましょう。まだまだ夜は冷え込む北海道からでした。

これが完成したゲームだ

こちらはサンプルでついてくるゲーム

ボスキャラを作り上げる

最後にスクロール速度や音楽設定をする

# 好きな街を好きな地形で

Kaneko Shuniti
金子 俊一

去年発売されて好評を博した「シムシティー」。この「テレインエディター」はその「シムシティー」のゲームの舞台となる地形を自由に作り出すことができるツールだ。これなら人口100万人も夢じゃないかな?

最初に断っておきたいのは，このソフトはゲームではなくてエディタであるってこと。シムシティーで，

「Now,Terrafoming」

ってメッセージを見たことあるでしょ。アレですよアレ。最初に「START NEW CITY」を選択すると出てくるやつね。大地や河川を作る，天地創造エディタってとこかな。

この「テレインエディター」だけではシムシティーはできない。すでに持っている人が利用するアプリケーションなのだ。

## スクロールなしよ◆◆◆◆◆◆◆◆◆◆◆

ゲームが立ち上がると，お馴染みの画面にマップウィンドウとエディットウィンドウが開いている。試しにシムシティーで作った街のデータをロードしてみる。

おんや？　ウィンドウの処理がヘンだぞ。まっ，気にしないか。さて，マップウィンドウを右端に持っていって……，と。

あれ？　マウスカーソルを右端に持っていってもスクロールしない。なんで？

人によって遊び方は変わるものだろうが，「私のシムシティー」では，スクロールする画面をいっぱいに使ってウィンドウを開いていた。カーソルを合わせてクリックするよりも，一気に右にカーソルを持っていくほうが何も考えずにできるからね。

出鼻をくじかれた思い。

## 掟破りの左クリック◆◆◆◆◆◆◆◆◆

やっぱ，基本は全体図でしょう。マップウィンドウを開く。「よし，ここいらへんをエディットしてやろう」。左クリック。

ありゃりゃ？　四角いワクが移動しない。シムシティーと操作が違うでないの。マップウィンドウで無意識に左クリックをするのは当たり前じゃない？

正解は右クリック。シムシティーに慣れている人ほど戸惑うだろう。ちなみに左クリックはエディットできてしまう。ボタンが逆ならまだよかったのにと思う。

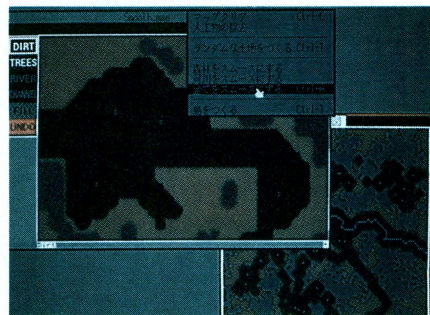
エディットウィンドウを開いてみる。新地，森林，河川を選択し，街を作っていく。船の航路もエディットできる。

しかし，1つひとつ地形を置いていくことしかできないので，これでエディットしろというのはグラフィックエディタのルーペモードしかないのと同じ。ドット打ちだけしかできないようなもの……。

## これは使える◆◆◆◆◆◆◆◆◆◆◆◆◆

シムシティーで使っていたデータをロードすると，街がそのままあらわれる。家や商店街もそのままなんだね。車が走っているままなのは驚いたけど，エディットするときにはこの街自体がじゃまになってくることもあるだろう。そんなときに便利なのが「人工物の除去」コマンド。これで人工物はすべて取り除ける。

ランダムな地形を作る，ようするに普通のテレイン機能もある。しかし，ちょっと違うのが，森林，湖沼の割合，河川の曲がり具合を調整できること。水の近くは人が住みやすいから水を多くとか，誰もが考えるよね。それを比率で作れるのはうれしい。

あと，閉じられた部分のFILLができる。これはPAINTと同じで，少しでも開いていると洩れてしまうし，遅いのが少し難点。

そのためか，1コマンドごとのアンドゥができるのはうれしい。

作り上げたばかりの地形はガタガタになっている。これを直すのがスムージング機能。角をとったり，より本物らしい地形にしてくれる。シムシティーのテレイン機能は最初からスムージングされてたんだね。

まあ，使いやすさはともかくとして，このツールを使うことで，できるだけ小さい土地で人口何万人を目指すとか，エディット画面1枚分で人口を何人まで増やせるか，といったような変わった遊び方ができるようになると思うよ。

スムージングしたあとの地形

とりあえず作ってみました

X68000用　5″2HD版　4,800円(税別)
イマジニア　☎03(3343)8911

### 操作体系が違うでしょ

シムシティーと操作法の統一がなされてないのは最大の失策。せっかく画面レイアウトを同じにしたのだから，徹底してほしい。

また，グラフィックエディタ並みとはいわずとも，ライン引きや部分コピー機能ぐらいは最低でもほしかった。欲をいえば，スキャナから取りこむ機能だってほしい。

家や道路，税金なども含めて街をエディットできればシナリオモードを自分で作れるような感じで楽しいと思う。家ばかりの街を作って，失業対策を考えたりとかね。○年○カ月後に災害が起こる指定ができるとか。うーん，シナリオエディタを作ったほうが売れるんじゃない？

**総合評価**

| 項目 | 0　　5　　10 |
|---|---|
| システム互換性 | ★★★ |
| 機能 | ★★★★★ |
| グラフィック | ★★★★★★ |
| マニュアル | ★★★★ |
| 創造性 | ★★★★★★★ |

# 詳説 System-7C

Furuhata Kazuhiro
古籏 一浩

MZ-700用ウィンドウシステム解説の続編です。先月の解説では全体的な構成が中心でした。今回はさらに詳しく内部構成を見ていきます。8ビットマシンでも十分稼動するシステムの秘密に迫ってみましょう。

## ウィンドウバッファ

ウィンドウバッファとはアプリケーションで使用するウィンドウ情報を列記したもので，20バイトで1個のウィンドウ情報を表します。エンドコードは−1です。このウィンドウバッファはIXレジスタによって参照することができます。表4の左側の番号はIXレジスタからのオフセットを示していますので，LD A, (IX+4) とするとウィンドウのY1座標（上）を読み出すことができます。

ウィンドウバッファのOn, Off Flagはウィンドウの状態を表しています。On, Off Flagと書いてありますが，0を書いてもウィンドウは表示されたままです。このフラグはウィンドウがイネーブル（有効）であるかディスエーブル（無効）であるかを示すものなのです（ほかの名前にすればよかったかな）。ウィンドウを見えなくするには，優先順位（IX+12）に0を書き込みます。この優先順位は1がいちばん手前で，32がいちばん奥となっています。

ほかのウィンドウシステムではどうやってウィンドウの優先順位を変えているのかよくわかりません。ハンドルを並べ替えているんでしょうか？　この優先順位の方法はスペースハリアー（正確にいうとメトロクロス）からいただきました。速い，簡単，メモリを消費しないといいことずくめです。

次にウィンドウIDですがこれは，ウィンドウごとに別々にしておいてください。

ウィンドウ座標は画面の左上隅（従来どおり）からです。実はウィンドウなどは仮想画面にすべて表示しておき，まとめて画面に転送しています。変棚，じゃない (Macintoshもこういう変換するのね)，変だなと思った人がいるかもしれませんね（いないだろーな)。本来ならばウィンドウ座標は仮想画面の座標そのものを示していたほうがスピードなどで有利なはずです。でも，アプリケーションを作成していくと，仮想画面の座標を示していると不都合なこともあるのです。ほかにも別のマシンに移植された場合のことも考えてこうなっているわけです。仮想画面のどこが原点となっているかは，SET.WINDOW.RECORDを呼び出したあとにHLレジスタにXY座標が入ります。

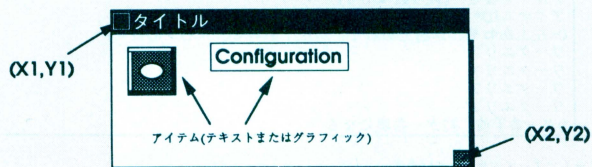
次のセレクトカラーはアイテムが選択された（されている）ときに背景となる色です。$44_H$とか$33_H$などに設定しておけばいいでしょう（$77_H$はだめです。白だから)。

次にアイテムタイプですが，ウィンドウマネージャはこのアイテムタイプを参照し，アイテムバッファのなかから，このウィンドウに対応したアイテムをサーチします。アイテムタイプは1から127までの値にしておいてください。この原理を利用すると，あるウィンドウに表示されているアイテムをそっくりそのままコピー（ではないが）して表示できます。さらにそのコピー（じゃないんですが）したアイテムを操作すると，コピー元（？）のウィンドウに表示されているアイテムも同様の操作がされたことになります。でも，危険だからあまりやらないほうがよいと思いますが。実はこんなことができるなんてたったいま思いついたのです（自分で作っておきながら)。

次のアイテム表示アドレスですが，これはシステムに定義されているアイテムや，ユーザーが定義したアイテムではどうしても表示できないときに使用します。表示させるルーチンがXPRINTだとすると，DW XPRINTのようにアセンブラで書いておけば自動的にXPRINTをウィンドウマネージャが呼び出します。注意としてはXPRINTのなかでは絶対にIX, IYを破壊してはいけません。ちなみにここが0000H

図1

**MENU&WINDOW　BUFFER**

| MENU& WINDOW BUFFER | | |
|---|---|---|
| 0 | On, Off Flag | ウィンドウの有無　(-1=エンドコード、0=無効、1=有効) |
| 1 | (Group) | (グループ番号、通常1) |
| 2 | ID | ウィンドウ別ID |
| 3 | (Window Type) | Bit0=TitleBar, Bit1=MainField, Bit2=SideBar, Bit3=UnderBar,Bit7=TitleBarSW |
| 4 | オフセットY1 | ウィンドウの上のオフセットy座標 |
| 5 | オフセットX1 | ウィンドウの左のオフセットx座標 |
| 6 | オフセットY2 | ウィンドウの下のオフセットy座標 |
| 7 | オフセットX2 | ウィンドウの右のオフセットx座標 |
| 8 | ---------- | 予備 |
| 9 | ---------- | 予備 |
| 10 | ---------- | 予備 |
| 11 | セレクトカラー | 選択されているアイテムの色 |
| 12 | 優先順位 | 1以上　(1=イネーブル、1以外はディスエーブル、0=ウィンドウを隠す) |
| 13 | アイテムタイプ | アイテムコード　負の数は不可 |
| 14 | Jump Address L | アイテム表示ジャンプアドレス下位　(Address=0000Hの時にはジャンプしない) |
| 15 | Jump Address H | アイテム表示ジャンプアドレス上位　(Address=0000Hの時にはジャンプしない) |
| 16 | Operation Adrs L | IDジャンプアドレス下位 |
| 17 | Operation Adrs H | IDジャンプアドレス上位 |
| 18 | ---------- | 予備 |
| 19 | ---------- | 予備 |

になっている場合，ウィンドウマネージャはなにもせず次の処理を行います。

さて最後のIDジャンプアドレスです。これはアイテムが選択された場合に，ID番号により分岐するアドレスのテーブルのアドレスです（以下の例の場合，DW IDJPTBL)。注意点はID＝0もこのテーブルに含まれることです。したがってテーブルの最初のジャンプアドレスは必ず0000Hになります。たとえば，

```
IDJPTBL：
        DW 0       ；必ず0
        DW ITEM1   ；ID＝1を持つアイ
                     テムの処理アドレス
        DW ITEM2   ；ID＝2を持つアイ
                     テムの処理アドレス
        DW ITEM3   ；ID＝3を持つアイ
                     テムの処理アドレス
              ⋮
```

## アイテムバッファ

図2を見てください。ウィンドウバッファはIXレジスタで参照できましたが，アイテムバッファはIYレジスタにより参照することができます。つまり，LD A，(IY＋9) でアイテムのIDを読み出すことができるわけです。

表の左側の番号はIYレジスタからのオフセットを表しています。まず，表示/無表示フラグですが，0にすると画面上からアイテムが消え，なおかつ判定（IDによる分岐も）もなくなります。1にすると通常の表示となり，2にすると選択された状態(ウィンドウバッファで設定したセレクトカラー）で表示されます。ただし，選択された場合でも，背景が透過になっていないと，普通の状態と変わりありません(少し注意)。次の表示タイプは，0の場合はグラフィック，1の場合はテキスト左詰め，2の場合はテキスト右詰め表示とします。1個のアイテムにつきひとつの属性しか設定することができません。この設定が表示タイプというわけです。

次の座標ですが，必ずX1≦X2，Y1≦Y2でなければなりません。この座標はウィンドウの左上（ポイントによって変わる）からの位置になっています。つまり，ほかのウィンドウシステムでいうローカル座標ということです。SYSTEM-7Cではアイテムの座標はローカル座標のローカル座標になるんでしょうね。

次にデータコードですが，ウィンドウマネージャのデータコードを見て，データバッファのなかから対応するデータをサーチするわけです。

次にグループ番号とIDです。グループ番号はウィンドウバッファのIDと同じにしておいてください。IDは重要です。ウィンドウマネージャはクリックされたときにどのウィンドウのどのアイテムが選択されたかを調べます。そしてサーチしたアイテムのID番号により，そのアイテムの処理が書いてあるサブルーチンへとジャンプします。IDが正の値だった場合は，ユーザーアイテムとしてウィンドウバッファで定義されたテーブルを参照しジャンプします。負の値だった場合は，システムで定義されているアイテムとして内部で処理を行い，場合によってはユーザーのサブルーチンへ処理を移します。

次にポイントフラグですが，これは原点をウィンドウのどこにするのかを設定します。通常0にしておけばまず，困ることはないと思います。最後のZ座標はウィンドウバッファの優先順位とほぼ一緒です。唯一異なるのは，0の場合がいちばん手前になるというところだけです。つまり，0が手前で32がいちばん奥になるわけです。

いま説明したのはユーザーアイテムの場合で，システムで定義されたアイテムは，ほかにも異なる定義をしてあるのがたくさんあります。ここではID＝−4のテキストエディットアイテムについて説明します。

標準アイテムと異なっているのは，表示タイプとデータコード，そしてIY＋11のワークエリアです。データコードはテキストエディットアイテムではアドレスを示すことになります。表示タイプは次の5つから表示形式を選択できます。

0　アドレスで示される40文字以内の文字列の修正　(エンドコード1AH)
1　アドレス内容の10進2桁表示
2　アドレス内容の10進4桁表示
3　アドレス内容の16進2桁表示
　(先頭に$マークがつきます)
4　アドレス内容の16進4桁表示
　(先頭に$マークがつきます)

IY＋11のワークエリアは，エディットフラグで，0のときには修正可能，1のときには修正不可（ロックされる）になります。このアイテムが選択されたときの文字の入力，入力内容の表示などはシステムが自動的に行いますので，ユーザーはただこのテキストエディットアイテムを設定するだけです。SX-WINDOWと比べて簡単でしょう。まあ，簡単なだけあって1行の入力しかできませんが。メモリがあればもっといいのができるんですけどね。

入力は通常どおりに行えます。ただし，小文字入力に関しては基本的にはできません。英数モードでシフト＋@でコントロールコードの05Hが入るようになっているので，むりやり入力できないこともありません。表示が変になるのと，入力文字数の関係上，あまりおすすめできません。

テキストエディットは文字入力に便利です。その他にもラジオボタン（複数アイテムのなかからひとつだけ選択可能）やチェックマーク（複数選択可能）など，便利なものが定義されています。

ほかのシステムアイテムについては説明書を読んでもらうしかありません（とてもOh!X誌上ですべて説明できない)。

DW ITEMn　；ID＝nを持つアイテムの処理アドレス

のようになります。

## データバッファ

アイテムバッファで説明したデータコードにより参照されるバッファがこれです。

図2

**ITEM BUFFER**

| ITEM BUFFER | | |
|---|---|---|
| 0 | 表示、無表示Flag | -1=エンドコード、0=無表示、1=表示、2=選択されている |
| 1 | 表示タイプ | 0=グラフィック、1=左づめ表示、2=中央表示、3=右づめ表示 |
| 2 | オフセットY1 | |
| 3 | オフセットX1 | 座標は必ず正の数であること |
| 4 | オフセットY2 | (テキスト表示時はビューエリア) |
| 5 | オフセットX2 | |
| 6 | Data Code 1 | テキスト、グラフィックのデータコード1(TextEdit時は文字列アドレス下位) |
| 7 | Data Code 2 | テキスト、グラフィックのデータコード2(TextEdit時は文字列アドレス下位) |
| 8 | グループ番号 | グループ番号　(負の数は不可) |
| 9 | ID | アイテムID* |
| 10 | ポイントフラグ | 0=左上あわせ、1=右上あわせ、2=左下あわせ、3=右下あわせ) |
| 11 | ワークエリア | ワークエリア |
| 12 | ワークエリア | ワークエリア |
| 13 | ワークエリア | ワークエリア |
| 14 | ワークエリア | ワークエリア |
| 15 | Z座標 | 0が一番手前、32が一番奥になる |

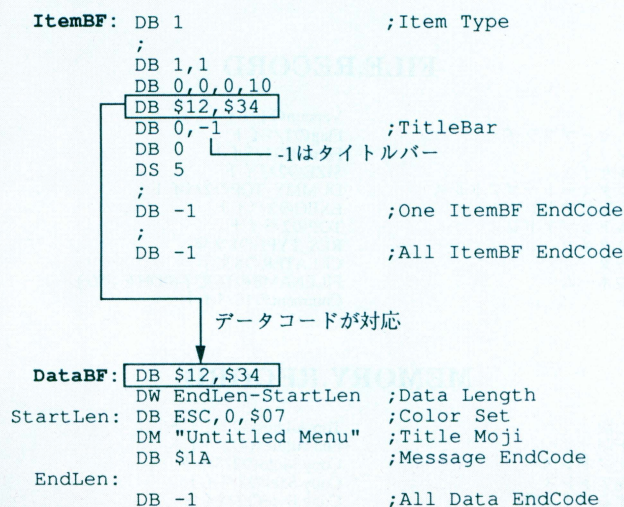
*アイテムIDで負数はシステムによって定義されています。

アイテムバッファとの対応は表3を見てください。データバッファはテキストまたはグラフィックデータがまとまったもので，最初に，データコード，次にデータ本体の長さ，そしてデータ本体がきます。エンドコードは−1になっています。注意する点はデータコードはこのデータ本体のあるアドレスを示すものではないということです。これは，アイテムバッファとデータバッファはバッファの先頭アドレスさえわかれば，メモリのどこにでも配置できるようにしてあるためです。もしデータコードがアドレスを示すとリロケータブルにすることができなくなってしまいます。ちなみにウィンドウバッファもリロケータブルになっています。

*　*　*

やっと，ウィンドウマネージャの説明が終わりました。説明が長かったわりにプログラムサイズは小さくて約4Kバイトです。ウィンドウを表示するだけならば，System-7Bにも似たようなルーチンがありました。System-7BのウィンドウはPrint文だけで表示させていたので，すごく遅くてウィンドウを移動させるのに，1秒くらいかかってしまうという，困ったルーチンでした。ちなみにSYSTEM-7Cのウィンドウ表示は，結構速いのでご安心を。

## メモリ関連その他

メモリマネージャはSYSTEM-7Cにはありません。が，サブルーチンとしてウィンドウマネージャに入っています。このルーチンは基本的にメモリのコピーを行います。移動も一種のコピーなので，コピー機能さえあればどうにでもなるはずです。このルーチンを使用する場合は，HLレジスタに次のテーブルアドレス(MEMTBL)を入れ，ファンクション番号の12をコールします。

```
MEMTBL:
    DW  MEMTOP ;メモリの先頭アドレス
    DW  MEMEND;メモリのエンドアドレス
    DW  COPYMOTO ;コピー元のアドレス
    DW  COPYSAKI ;コピー先のアドレス
    DW  COPYBYTE ;コピーするバイト数
```

正常に動作した場合は，CY=0（ノンキャリー）で戻ってきます。エラーがあった場合は，CY=1（キャリー）で戻ってきます。エラーは次のような場合です。

★コピー元またはコピー先のアドレスがMEMTOPとMEMENDからはみ出している場合。

★コピーしたものが，MEMTOPとMEMENDからはみ出す場合（事前チェックのこと）。

**●ロード&セーブモジュール**

カセットの書き込み，読み込みを行います(MZフォーマット/1200ボー)。近いうちにQDに対応する予定です。このプログラムはサイズを小さくするためにROMモニタにほとんど頼りきっています。

**●サービスルーチン**

その名のごとくサービスルーチンです。現在のところ，ファイルのロードセーブのためのものしか入っていません。まだかなりメモリに余裕があるので，いろいろ追加する予定です。

この原稿を書いている時点では，このサービスルーチンはSYSTEM-7C Ver.0.7のもので動作していたものです。手直ししなくても動くので，そのままにしてあります（相当な無駄があります）。

## その他のマネージャ/レコード

ピクチャードライバは標準フォーマットでグラフィックを描画します。System.Configは主にキーマップの設定です。System.Configのキーマップを変えればオリジナルのキーマップにすることができます。だいたいこんなところです。メモリに余裕があればほかにもいろいろ入れられましたが，まあしょうがないですね。あと，プリンティングマネージャがありません。だいたい本人がプリンタを持っていないという理由もあります。

**●ドラッグレコード**

このドラッグレコードはウィンドウマネージャに入っています。このルーチンを使用すると，ドラッグ/リサイズの処理を簡単に行うことができます。レコード内容は表1のようになっています。ドラッグレコードで設定する座標はすべて絶対座標（仮想画面の座標）で指定しますので注意してください。まず，Y, X座標ですが，この座標は現在のカーソル座標です。次のY1,X1,Y2,X2はドラッグまたはリサイズするものの座標です。ウィンドウをドラッグする場合ならば，これらの座標はウィンドウの左上と右下になります。

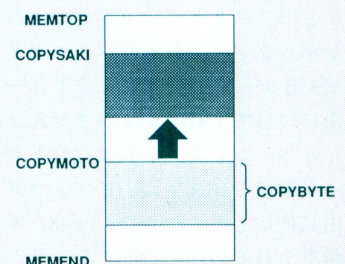
表示タイプは，反転表示するときの図形です。ドラッグ/リサイズフラグはドラッグなのか，リサイズなのかを示すフラグです。“0”のときドラッグ，“1”のときリサイズになります。次の2バイトはウィンドウマネー

図3

### アイテムバッファ、データバッファの対応

```
ItemBF: DB 1                    ;Item Type
        ;
        DB 1,1
        DB 0,0,0,10
        DB $12,$34
        DB 0,-1                 ;TitleBar
        DB 0     └── -1はタイトルバー
        DS 5
        ;
        DB -1                   ;One ItemBF EndCode
        ;
        DB -1                   ;All ItemBF EndCode

              データコードが対応

DataBF: DB $12,$34
        DW EndLen-StartLen      ;Data Length
StartLen: DB ESC,0,$07          ;Color Set
        DM "Untitled Menu"      ;Title Moji
        DB $1A                  ;Message EndCode
 EndLen:
        DB -1                   ;All Data EndCode
```

図4

### メモリーのコピー/&移動

ジャが返す値で，ドラッグの場合，移動量が入ります。リサイズの場合は変更量が入ります。最後のコンディションコールアドレスは，リサイズ時に制限をしたい場合などに使用します。その場合，IYレジスタに内部ドラッグレコードのアドレスが入ります。内部ドラッグレコードも，いま説明したものと変わりませんので，LD A，(IY)とすることでY座標が読み出せます。

ボタンが離されるとウィンドウマネージャはドラッグレコードに値を入れて戻ってきます。この場合も戻ってきた座標は絶対座標なので注意してください。

**●ファイルレコード**

これは，ロード/セーブモジュールに内蔵されています。ロード/セーブモジュールに値を渡す場合は，HLレジスタにレコードアドレスを入れ，システムコールの161をコールします。では，ファイルレコードについて説明しましょう。まず，バージョン番号は1にしてください。ロード/セーブフラグは，

| | |
|---|---|
| 0 | ヘッダロード |
| 1 | データロード |
| 2 | ヘッダとデータ両方ロード |
| 3 | ヘッダセーブ |
| 4 | データセーブ |
| 5 | ヘッダとデータ両方セーブ |

のようになっていますので処理したい番号を設定してください。

フォーマットは1または0を設定してください。SIZEはロード/セーブするバイト数です。DUMMYTOPアドレスは，ロードセーブするデータのアドレスです。EXECは実行アドレスです。TOPアドレスはロード/セーブモジュールを使用した場合にロード/セーブする先頭アドレスです。これを利用するとZEDAなどのオフセットロード/オフセットセーブができます。次のRES.TYPE（リソースタイプ）ですが，これはASCIIコード8文字で，セーブされているデータのタイプを示します。現在なにもタイプが設定されていませんので自由に使っていただいて結構です。

次のCREATERはセーブされているデータがどのアプリケーションで作られたものかを示します。これもリソースタイプ同様，ASCIIコード8文字です。クリエータ名も自由に付けてくれてかまいません。次はファイルネームです。これは従来どおり改行コードを含む16文字以内です。コメントは自由に使用してください。先頭8バイトには日付を入れるのが一般的でしょうか。別に入れなくてもかまいません。

表1

## WINDOW RECORD

| | |
|---|---|
| バージョン | Versionの1バイト |
| ユーザーが使用しているXYのアドレス | Y, Xの2バイト |
| スクリーンビューの大きさ | Y1, X1, Y2, X2の4バイト(オフセット) |
| ウィンドウバッファのアドレス | Low, Highの2バイト |
| アイテムバッファのアドレス | Low, Highの2バイト |
| データバッファのアドレス | Low, highの2バイト |
| スクリーン表示時のオフセット座標 | Y, Xの2バイト |
| 表示スクリーン(VRAM)の座標 | Y, Xの2バイト |
| 表示スクリーン(VRAM)の長さ | LYの1バイト |
| 外部イベントのアドレス | Low, Highの2バイト(0の時はパスコード) |
| Zeroトラップ時のジャンプアドレス | Low, Highの2バイト(0の時はパスコード) |
| 予備 | 0000H |

## WINDOW RECORD.EXTEND

| | |
|---|---|
| スクリーンのベースXY | Y, Xの2バイト |
| バックグランドのキャラクタと色 | CHR, COLORの2バイト |
| ウィンドウのイネーブルタイトル色 | COLORの1バイト |
| ウィンドウのディスエーブルタイトル色 | COLORの1バイト |
| ウィンドウ内のキャラクタ | CHRの1バイト |
| ウィンドウの右枠のキャラクタ | CHRの1バイト |
| ウィンドウの下枠のキャラクタ | CHRの1バイト |
| ウィンドウの中の色 | COLORの1バイト |
| ウィンドウの影の色 | COLORの1バイト |
| アイテムの点滅回数 | FLASHの1バイト |
| ウィンドウリサイズ時の最小幅 | MinY, MinXの2バイト |
| ウィンドウリサイズ時の最大幅 | MaxY, MaxXの2バイト |
| チェックマークのキャラクタと色 | CHR, COLORの2バイト |

## DRAG RECORD

| | |
|---|---|
| 現在のカーソル座標 | Y, X |
| 最初の座標 | Y1, X1, Y2, X2 |
| 表示タイプ | ライン、ボックス、サークル(1バイト) |
| ドラッグ、リサイズフラグ | 0=ドラッグ、1=リサイズ(1バイト) |
| リターン用ワーク(移動量) | 移動した量(Y, Xの2バイト) |
| コンディションコールアドレス | Low, High (引数はIY=DRAG RECORD) |

## EVENT RECORD

| | |
|---|---|
| バージョン | Versionの1バイト |
| カーソルキャラクタ | CHR1, COLOR1, CHR2, COLOR2の4バイト |
| 点滅形式 | Flash Conditionの1バイト |
| キーボードリピート | Repeatの2バイト |
| 予備 | 0000H |
| TEXT.EDIT用キーリピート | Repeatの2バイト |
| TEXT.EDIT用ワークエリアのアドレス | Addressの2バイト |
| 標準のキーテーブルアドレス | Addressの2バイト |
| シフトキーのキーテーブルアドレス | Addressの2バイト |
| カナキーのキーテーブルアドレス | Addressの2バイト |
| グラフキーのキーテーブルアドレス | Addressの2バイト |

## EDIT.RECORD

| | |
|---|---|
| XY座標 | XYの2バイト |
| 左リミットX，右リミットX | LMT-X, LMT-Yの2バイト |
| エディットアドレス | Addressの2バイト |
| 文字列表示実行ルーチンアドレス | Addressの2バイト |

## FILE.RECORD

| | |
|---|---|
| バージョン | Versionの1バイト |
| ロード、セーブフラグ | Flagの1バイト |
| フォーマット | Formatの1バイト |
| ファイルサイズ | SIZEの2バイト |
| ファイルダミートップアドレス | DUMMY_TOPの2バイト |
| ファイル実行アドレス | EXECの2バイト |
| ファイルトップアドレス | TOPの2バイト |
| リソースタイプ | RES_TYPEの8文字 |
| クリエータ | CREATERの8文字 |
| ファイルネーム | FILENAMEの16文字(0DHを含む) |
| コメント | Commentの16バイト |

## MEMORY.RECORD

| | |
|---|---|
| メモリ上限 | Top Addressの2バイト |
| メモリ下限 | End Addressの2バイト |
| コピー元アドレス | Copy Motoの2バイト |
| コピー先アドレス | Copy Sakiの2バイト |
| コピーするバイト数 | Copy Byteの2バイト |

## プログラミング

さて，プログラムのお時間がやってまいりました。ここではSYSTEM-7Cでプログラムを組む人のための解説です。

SYSTEM-7Cの場合，フォントマネージャ以降の上層ルーチンと下層ルーチンとではプログラムスタイルが異なっています。システムコール一覧を見るとフォントマネージャ以降は極端にコール数が少なくなっていますが，カーネル，スペースグラフィックはコール数が多くなっているのがわかると思います。これは，カーネルなどはルーチン1つひとつが独立しており，レジスタにパラメータを入れてコールするという従来の方式をとっているのに対し，マネージャ群は初めにレコードを設定し，あとはシステムがレコードを参照し処理していくようになっているためです。

マネージャ群は，最初にレコードを設定しておかないとほかのマネージャ内のルーチンが正常に動作しません。マネージャ群を使用する場合はレコード設定をお忘れなく。レコードの設定方法は共通で，Aレジスタに0または1，HLレジスタに設定レコードアドレスを入れ，SET.~.RECORDをコールします。

カーネル，スペースグラフィックはSystem-7Bのルーチンを改良したものがほとんどです。したがってレジスタ渡しのものがほとんどです。しかし，いちいちルーチンごとにパラメータを覚えるのは大変です。そこで似たような動作をするものはなるべく同じパラメータにしてあります。たとえば，ライン，ボックス(フィル)，サークル(フィル)，スクロールは，X1=H-reg，Y1=L-reg，X2=D-reg，Y2=E-regのようにしてあります。PUTルーチンもX=H-reg，Y=L-reg，DE=データアドレス，B=横の長さ，C=縦の長さのようになっています。

また，SYSTEM-7Cはシングルタスク*1なので，メモリは自由に使っても構わないのですが，できれば使用メモリもプログラム側のサイズに含めてもらえるとありがたいですね。なぜかというと，ソースリストが配布される関係上，個人で自由なアドレスにアセンブルして複数のソフトをメモリ内に持つ可能性があるからです。オフスクリーンバッファが必要なソフトの場合もできれば，メモリをプログラム内で確保してほしいところです。だめな場合はちゃんとワークエリアを明記しておきましょう。

表2

### System ID

| ID=0 (InRect) | | |
|---|---|---|
| 0 | 表示、無表示Flag | 無視される |
| 1 | 表示タイプ | 無視される |
| 2 | オフセットY1 | 無視される |
| 3 | オフセットX1 | 無視される |
| 4 | オフセットY2 | 無視される |
| 5 | オフセットX2 | 無視される |
| 6 | Data Code 1 | 無視される |
| 7 | Data Code 2 | 無視される |
| 8 | グループ番号 | 無視される |
| 9 | ID | 0 |
| 10 | ポイントフラグ | 無視される |
| 11 | ワークエリア | 無視される |
| 12 | ワークエリア | 無視される |
| 13 | ワークエリア | 無視される |
| 14 | ワークエリア | 無視される |
| 15 | Z座標 | 0～31 |

| ID=-1 (TitleBar) | | |
|---|---|---|
| 0 | 表示、無表示Flag | 1 |
| 1 | 表示タイプ | 1, 2, 3 |
| 2 | オフセットY1 | 自動設定 |
| 3 | オフセットX1 | 0か1 |
| 4 | オフセットY2 | 自動設定 |
| 5 | オフセットX2 | 自動設定 |
| 6 | Data Code 1 | 文字列Code |
| 7 | Data Code 2 | 文字列Code |
| 8 | グループ番号 | Free (0~127) |
| 9 | ID | -1 |
| 10 | ポイントフラグ | 0 |
| 11 | ---------- | 0 |
| 12 | ワークエリア | Free |
| 13 | ワークエリア | Free |
| 14 | ワークエリア | Free |
| 15 | Z座標 | 0～31 |

| ID=-2 (CloseBox) | | |
|---|---|---|
| 0 | 表示、無表示Flag | 1 |
| 1 | 表示タイプ | 無視される |
| 2 | オフセットY1 | 0 |
| 3 | オフセットX1 | 0 |
| 4 | オフセットY2 | 0 |
| 5 | オフセットX2 | 0 |
| 6 | Data Code 1 | 無視される |
| 7 | Data Code 2 | 無視される |
| 8 | グループ番号 | Free (0~127) |
| 9 | ID | -2 |
| 10 | ポイントフラグ | 0 |
| 11 | ワークエリア | Free |
| 12 | ワークエリア | Free |
| 13 | ワークエリア | Free |
| 14 | ワークエリア | Free |
| 15 | Z座標 | 0～31 |

| ID=-3 (SizeBox) | | |
|---|---|---|
| 0 | 表示、無表示Flag | 1 |
| 1 | 表示タイプ | 無視される |
| 2 | オフセットY1 | 0 |
| 3 | オフセットX1 | 0 |
| 4 | オフセットY2 | 0 |
| 5 | オフセットX2 | 0 |
| 6 | Data Code 1 | 無視される |
| 7 | Data Code 2 | 無視される |
| 8 | グループ番号 | Free (0~127) |
| 9 | ID | -3 |
| 10 | ポイントフラグ | 3 |
| 11 | ワークエリア | Free |
| 12 | ワークエリア | Free |
| 13 | ワークエリア | Free |
| 14 | ワークエリア | Free |
| 15 | Z座標 | 0～31 |

| ID=-4 (TextBox) | | |
|---|---|---|
| 0 | 表示、無表示Flag | 1 |
| 1 | 表示タイプ | *** |
| 2 | オフセットY1 | Free |
| 3 | オフセットX1 | Free |
| 4 | オフセットY2 | 無視される |
| 5 | オフセットX2 | Free |
| 6 | Data Code 1 | TextAdrs L |
| 7 | Data Code 2 | TextAdrs H |
| 8 | グループ番号 | Free (0~127) |
| 9 | ID | -4 |
| 10 | ポイントフラグ | 0~3 |
| 11 | EditFlag | 0=Ok, 1=No |
| 12 | ワークエリア | Free |
| 13 | ワークエリア | Free |
| 14 | ワークエリア | Free |
| 15 | Z座標 | 0～31 |

| ID=-5 (SideLine) | | |
|---|---|---|
| 0 | 表示、無表示Flag | 1 |
| 1 | 表示タイプ | 無視される |
| 2 | オフセットY1 | Free |
| 3 | オフセットX1 | Free |
| 4 | オフセットY2 | 無視される |
| 5 | オフセットX2 | Free |
| 6 | Data Code 1 | 無視される |
| 7 | Data Code 2 | 無視される |
| 8 | グループ番号 | Free (0~127) |
| 9 | ID | -5 |
| 10 | ポイントフラグ | 0~3 |
| 11 | ワークエリア | Free |
| 12 | ワークエリア | Free |
| 13 | ワークエリア | Free |
| 14 | ワークエリア | Free |
| 15 | Z座標 | 0～31 |

*** 0=文字列
1=10進2桁
2=10進4桁
3=16進2桁
4=16進4桁

| ID=-6 (ChrList) | | |
|---|---|---|
| 0 | 表示、無表示Flag | 1 |
| 1 | 表示タイプ | 無視される |
| 2 | オフセットY1 | Free |
| 3 | オフセットX1 | Free |
| 4 | オフセットY2 | Y1+15 |
| 5 | オフセットX2 | X1+15 |
| 6 | Data Code 1 | JumpAdrs L |
| 7 | Data Code 2 | JumpAdrs H |
| 8 | グループ番号 | Free (0~127) |
| 9 | ID | -6 |
| 10 | ポイントフラグ | 0~3 |
| 11 | 選択垂直位置 | Vertical |
| 12 | 選択水平位置 | Horizontal |
| 13 | ワークエリア | Free |
| 14 | ワークエリア | Free |
| 15 | Z座標 | 0～31 |

*JumpAdrs=0の時はジャンプしない

| ID=-7 (ColorList) | | |
|---|---|---|
| 0 | 表示、無表示Flag | 1 |
| 1 | 表示タイプ | 無視される |
| 2 | オフセットY1 | Free |
| 3 | オフセットX1 | Free |
| 4 | オフセットY2 | Y1+15 |
| 5 | オフセットX2 | X1+15 |
| 6 | Data Code 1 | JumpAdrs L |
| 7 | Data Code 2 | JumpAdrs H |
| 8 | グループ番号 | Free (0~127) |
| 9 | ID | -7 |
| 10 | ポイントフラグ | 0~3 |
| 11 | 選択垂直位置 | Vertical |
| 12 | 選択水平位置 | Horizontal |
| 13 | ワークエリア | Free |
| 14 | ワークエリア | Free |
| 15 | Z座標 | 0～31 |

*JumpAdrs=0の時はジャンプしない

| ID=-8 (RadioButton) | | |
|---|---|---|
| 0 | 表示、無表示Flag | 1 |
| 1 | 表示タイプ | 1 |
| 2 | オフセットY1 | Free |
| 3 | オフセットX1 | Free |
| 4 | オフセットY2 | Free |
| 5 | オフセットX2 | Free |
| 6 | Data Code 1 | Free |
| 7 | Data Code 2 | Free |
| 8 | グループ番号 | Free (0~127) |
| 9 | ID | -8 |
| 10 | ポイントフラグ | 0~3 |
| 11 | 選択位置 | 0~Y 2 |
| 12 | JumpAdrs Low | JumpAdrs L |
| 13 | JumpAdrs High | JumpAdrs H |
| 14 | ワークエリア | Free |
| 15 | Z座標 | 0～31 |

*JumpAdrs=0の時はジャンプしない

| ID=-9 (CheckMark) | | |
|---|---|---|
| 0 | 表示、無表示Flag | 1 |
| 1 | 表示タイプ | 1 |
| 2 | オフセットY1 | Free |
| 3 | オフセットX1 | Free |
| 4 | オフセットY2 | Free |
| 5 | オフセットX2 | Free |
| 6 | Data Code 1 | Free |
| 7 | Data Code 2 | Free |
| 8 | グループ番号 | Free (0~127) |
| 9 | ID | -9 |
| 10 | ポイントフラグ | 0~3 |
| 11 | 選択ビット下位 | SelectBitLow |
| 12 | 選択ビット上位 | SelectBitHigh |
| 13 | JumpAdrs Low | JumpAdrs L |
| 14 | JumpAdrs High | JumpAdrs H |
| 15 | Z座標 | 0～31 |

*JumpAdrs=0の時はジャンプしない
選択ビットのビットが1のときは選択されており、0の時は選択されていない

| ID=-10 (CheckMark2) | | |
|---|---|---|
| 0 | 表示、無表示Flag | 1 |
| 1 | 表示タイプ | 1 |
| 2 | オフセットY1 | Free |
| 3 | オフセットX1 | Free |
| 4 | オフセットY2 | Free |
| 5 | オフセットX2 | Free |
| 6 | Data Code 1 | Free |
| 7 | Data Code 2 | Free |
| 8 | グループ番号 | Free (0~127) |
| 9 | ID | -10 |
| 10 | ポイントフラグ | 0~3 |
| 11 | 選択ビット下位 | SelectBitLow |
| 12 | 選択ビット上位 | SelectBitHigh |
| 13 | JumpAdrs Low | JumpAdrs L |
| 14 | JumpAdrs High | JumpAdrs H |
| 15 | Z座標 | 0～31 |

*JumpAdrs=0の時はジャンプしない
選択ビットのビットが1のときは選択されており、0の時は選択されていない

| ID=-11 (SizeChange) | | |
|---|---|---|
| 0 | 表示、無表示Flag | 1 |
| 1 | 表示タイプ | 無視される |
| 2 | オフセットY1 | 0 |
| 3 | オフセットX1 | 0 |
| 4 | オフセットY2 | 0 |
| 5 | オフセットX2 | 0 |
| 6 | Data Code 1 | Free |
| 7 | Data Code 2 | Free |
| 8 | グループ番号 | Free (0~127) |
| 9 | ID | -11 |
| 10 | ポイントフラグ | 1 |
| 11 | Change Y1 | Change Y1 |
| 12 | Change X1 | Change X2 |
| 13 | Change Y2 | Change Y2 |
| 14 | Change X2 | Change X2 |
| 15 | Z座標 | 0～31 |

| ID=-12 (TransBox) | | |
|---|---|---|
| 0 | 表示、無表示Flag | 1 |
| 1 | 表示タイプ | 0 |
| 2 | オフセットY1 | Free |
| 3 | オフセットX1 | Free |
| 4 | オフセットY2 | Free |
| 5 | オフセットX2 | Free |
| 6 | Data Code 1 | JumpAdrs L |
| 7 | Data Code 2 | JumpAdrs H |
| 8 | グループ番号 | Free (0~127) |
| 9 | ID | -12 |
| 10 | ポイントフラグ | 0～3 |
| 11 | ワークエリア | Free |
| 12 | ワークエリア | Free |
| 13 | ワークエリア | Free |
| 14 | ワークエリア | Free |
| 15 | Z座標 | 0～31 |

| ID=-13 (Boxfill) | | |
|---|---|---|
| 0 | 表示、無表示Flag | 1 |
| 1 | 表示タイプ | 1 |
| 2 | オフセットY1 | Free |
| 3 | オフセットX1 | Free |
| 4 | オフセットY2 | Free |
| 5 | オフセットX2 | Free |
| 6 | Data Code 1 | JumpAdrs L |
| 7 | Data Code 2 | JumpAdrs H |
| 8 | グループ番号 | Free (0~127) |
| 9 | ID | -13 |
| 10 | ポイントフラグ | 0～3 |
| 11 | フィル形式 | bit 0:chr,1:atb |
| 12 | フィルキャラクタ | Character |
| 13 | フィルカラー | Color |
| 14 | ワークエリア | Free |
| 15 | Z座標 | 0～31 |

| ID=-14 (PaintView) | | |
|---|---|---|
| 0 | 表示、無表示Flag | 1 |
| 1 | 表示タイプ | 1 |
| 2 | オフセットY1 | Free |
| 3 | オフセットX1 | Free |
| 4 | オフセットY2 | Free |
| 5 | オフセットX2 | Free |
| 6 | Data Code 1 | JumpAdrs L |
| 7 | Data Code 2 | JumpAdrs H |
| 8 | グループ番号 | Free (0~127) |
| 9 | ID | -14 |
| 10 | ポイントフラグ | 0～3 |
| 11 | ワークエリア | 0 |
| 12 | ワークエリア | 0 |
| 13 | ワークエリア | 0 |
| 14 | ワークエリア | 0 |
| 15 | Z座標 | 0～31 |

A=Push Bit, BC=DXY
(DE=Drag.Record)

| ID=-15 (Chr offscrnBF) | | |
|---|---|---|
| 0 | 表示、無表示Flag | 1 |
| 1 | 表示タイプ | MLX |
| 2 | オフセットY1 | Free |
| 3 | オフセットX1 | Free |
| 4 | オフセットY2 | Free |
| 5 | オフセットX2 | Free |
| 6 | Data Code 1 | JumpAdrs L |
| 7 | Data Code 2 | JumpAdrs H |
| 8 | グループ番号 | Free (0~127) |
| 9 | ID | -15 |
| 10 | ポイントフラグ | 0～3 |
| 11 | Scrn表示Y座標 | MY |
| 12 | Scrn表示X座標 | MX |
| 13 | Scrn表示長さY | ?LY |
| 14 | Scrn表示長さX | ?LX |
| 15 | Z座標 | 0～31 |

| ID=-16 (Atb offscrnBF) | | |
|---|---|---|
| 0 | 表示、無表示Flag | 1 |
| 1 | 表示タイプ | MLX |
| 2 | オフセットY1 | Free |
| 3 | オフセットX1 | Free |
| 4 | オフセットY2 | Free |
| 5 | オフセットX2 | Free |
| 6 | Data Code 1 | JumpAdrs L |
| 7 | Data Code 2 | JumpAdrs H |
| 8 | グループ番号 | Free (0~127) |
| 9 | ID | -16 |
| 10 | ポイントフラグ | 0～3 |
| 11 | Scrn表示Y座標 | MY |
| 12 | Scrn表示X座標 | MX |
| 13 | Scrn表示長さY | ?LY |
| 14 | Scrn表示長さX | ?LX |
| 15 | Z座標 | 0～31 |

| ID=-17 (Move Box) | | |
|---|---|---|
| 0 | 表示、無表示Flag | 1 |
| 1 | 表示タイプ | 0 |
| 2 | オフセットY1 | Free |
| 3 | オフセットX1 | Free |
| 4 | オフセットY2 | Free |
| 5 | オフセットX2 | Free |
| 6 | Data Code 1 | JumpAdrs L |
| 7 | Data Code 2 | JumpAdrs H |
| 8 | グループ番号 | Free (0~127) |
| 9 | ID | -17 |
| 10 | ポイントフラグ | 0～3 |
| 11 | ワークエリア | 2 |
| 12 | ワークエリア | 0 |
| 13 | 表示色 | Color |
| 14 | ワークエリア | 0 |
| 15 | Z座標 | 0～31 |

シングルタスクということで，メモリマネージャがありません。ただし，メモリのコピー（移動）は，ウィンドウマネージャ内のルーチンで処理可能です。また，マネージャ群はI/O，VRAMのD400$_{H}$～D7FF$_{H}$，DC00$_{H}$～DFFF$_{H}$を一時的なワークエリアとして使用しているので注意してください。

SYSTEM-7C上で動作するアプリケーションを作成する場合には，Window Builder (WB) を使用すると，比較的短期間でアプリケーションを作成することができます。

使用言語はアセンブラとなっていますが，パラメータを正しく渡せるならばなんでも構わないはずです（当然か）。S-OSは言語関係が豊富ですから自分の好きなものを使えばいいでしょう。

SYSTEM-7Cのウィンドウマネージャの場合，各アイテムが独立しているので，最悪アイテムの実行アドレスさえわかれば動作可能です。もし，Macintoshのファインダ，SX-WINDOWのShellみたいにしたいのならば，自分でアイコンを書いてアイテムとして登録（実行アドレスなども）します。

メモリ上にロードされているときは，アイテムの実行アドレスを登録し，ロードされていない場合は，実行アドレスを0000$_{H}$にしておけばできあがりです（実行アドレスが0000$_{H}$のときはなにも処理しません。メモリにロードされているかチェックするのは自前でやってください）。

---

＊1　つまり，基本的に一度にひとつのソフトしか動作させることができない。ついでにMacintoshのFinder，SX-WINDOWのShellに相当するものがありません。要求があれば作成するかもしれません。でも基本的にカセットが主体ですから……。

**表3**

| No. | | Name | Description |
|---|---|---|---|
| | | -------------------- JUMP TABLE -------------------- | |
| | | -------------------- OS KERNEL -------------------- | |
| 0 | • | INIT | (Initialize all subroutines) |
| 1 | • | Version | (Get Version Number) |
| 2 | • | Check 700/1500 | (Machine Check) |
| 3 | • | Close I/O | (V-RAM Close) |
| 4 | • | Open I/O | (V-RAM Open) |
| 5 | • | Set.KVRAM.address | (High-order Set) |
| 6 | • | Read.KRAM.address | (High-order Read) |
| 7 | • | TDXY | (Calculate X, Y -> KVRAM address) |
| 8 | • | VTDXY | (Calculate X, Y -> V-RAM address) |
| 9 | • | GET.XY of KVRAM | (Calculate V-RAM address -> X, Y) |
| 10 | • | Trans40xy | (Trans KVRAM -> V-RAM) |
| 11 | • | Dual Trans40xy | (Trans KVRAM -> V-RAM) |
| 12 | • | Laster Print | (High-speed 40chr Write) |
| 13 | • | Color convert | (Color Convert OLD.BF -> NEW.BF) |
| 14 | • | SET.GET.BF | (Use Polygon Fill) |
| 15 | • | Read.GET.BF | (Use Polygon Fill) |
| 16 | • | STTM | (Start Line, Enemy Beam) |
| 17 | • | MVTM | (Execute Line, Enemy Beam) |
| 18 | • | Print Score | (Print Score to KVRAM or V-RAM) |
| 19 | • | Score Plus | (Add the Score Buffer and BC-reg ) |
| 20 | • | SGN8Bit | (Get Sign of A-reg) |
| 21 | • | SET.PAUSE.MESSAGE | (Pause Message Process Address Set) |
| 22 | | | |
| 23 | • | PAUSE | (Graph key to Pause, any key to Restart) |
| 24 | • | SET.Key.Table | (Key Matrix -> Asc convert Table set) |
| 25 | • | Read.Key.Table | (Key Matrix -> Asc convert Table read) |
| 26 | • | SET.KEY.MASK | (Key Matrix Port Data Mask Table Set) |
| 27 | • | READ.KEY.MASK | (Key Matrix Port Data Mask Table Read) |
| 28 | • | GET.Key | (Get Key, ASCII Code) |
| 29 | • | GET.Key.Port | (Key Port Direct Read) |
| 30 | • | Read.Key.BF | (Key Buffer Direct Read) |
| 31 | • | Set GAME Key | (Left Hand or Right Hand Select) |
| 32 | • | Read GAME Key | (Left or Right Hand ? Data Read) |
| 33 | • | GAME.Key | (Read GAME Keys) |
| 34 | • | Move.Fighter | (Move.Fighter by GAME Key Code) |
| 35 | • | Hantei | (Check 2 Rectangle Match) |
| 36 | • | Hantei2 | (Check 1 point AREA) |
| 37 | • | SET.Random.Value | (Random Value Set) |
| 38 | | | |
| 39 | • | Random | (16 bit Random value) |
| 40 | • | XYCood | (Swap if X1>X2 or Y1>Y2) |
| 41 | • | XY.len | (X2-X1+1, Y2-Y1+1) |
| 42 | • | Bit.Map.OBJ | (Convert Bit Map to Pixel Map) |
| 43 | • | Bit.Map.OBJ.plus | (Add Memory and Conveted Bit.Map) |
| 44 | • | Print Teki Buffer | (Custom buffer print) |
| 45 | • | Hantei.Teki | (Check XY and TekiXY) |
| 46 | | | |
| 47 | | | |
| 48 | | | |
| 49 | • | GET.ASC | (BIN to ASC,HL=Value,DE=Buf8Byte) |
| 50 | • | ZAHYO | (Use Font Manager) |
| 51 | • | COLOR.ZAHYO | (Use Font Manager) |
| 52 | • | SET.VRAM.SW | (Use Font Manager) |
| 53 | • | READ.VRAM.SW | (Use Font Manager) |
| 54 | • | READ.SWITCH | (Joy Stick Port Read) |
| 55 | • | READ.STICK | (Joy Stick Switch Read) |
| 56 | • | Multi 8×8=16 | |
| 57 | • | Division 16+8=8...8 | |
| 58 | • | Multi16×16=32 | |
| 59 | • | Division 32+16=32...16 | |
| 60 | • | GET.VALUE.DEC | (In Event Manager) |
| 61 | • | GET.VALUE.HEX | (In Event Manager) |
| 62 | • | GAME.KEY.JUMP.ADDRESS (HAND>2) | |
| 63 | • | SET.TASK.RECORD | |
| 64 | • | READ.TASK.RECORD | |
| 65 | | CMS | |
| | | -------------------- SOUND ROUTINE -------------------- | (1B00H) |
| 67 | | START.SOUND | |
| 68 | | STOP.SOUND | |
| 69 | | | |
| 70 | | | |
| 71 | | READ.SOUND.RECORD | |
| 72 | | PRESS.GRAPHIC.DATA | |
| 73 | | OBJECT.GRAPHIC.DATA | |
| | | -------------------- SPACE GRAPHICS -------------------- | (1E00H) |
| 74 | • | SET.PEN.STYLE | (Graphic Pen Address Style set) |
| 75 | • | READ.PEN.STYLE | (Graphic Pen Address Style read) |
| 76 | • | SET.PEN.Put.Large&Small | (Put LS Pen Style Address set) |
| 77 | • | READ.PEN.Put.Large&Small | (Put LS Pen Style Address read) |
| 78 | • | SET.PEN.CIRCLE | (Circle, Circle.Fill Pen set) |
| 79 | • | READ.PEN.CIRCLE | (Circle, Circle.Fill Pen read) |
| 80 | | | |
| 81 | | | |
| 82 | • | GET | (Get Data KVRAM to Memory) |
| 83 | • | Put.chr | (Put Character data only) |
| 84 | • | Put.over.chr | (Put Character data & Mask) |
| 85 | • | Put.atb | (Pair Put.chr) |
| 86 | • | Put.atb.cpl | (Put Color data Reverse) |
| 87 | • | Put.over.atb | (Put Color data & Mask) |
| 88 | • | Put.over.atb.cpl | (Put Color data reverse & Mask) |
| 89 | • | Put.hyp | (High-speed Put) |
| 90 | • | Put.Large&Small | (Put Resize data) |
| 91 | • | Box.Normal | (Box write by direct code) |
| 92 | • | Box.Color | (Box write by color modify) |
| 93 | • | Box.chr | (Alart Box) |
| 94 | • | Box.atb | (Alart Box) |
| 95 | • | Box.Fill | (Box.Fill by direct code) |
| 96 | • | Box.PEN | (Use PEN) |
| 97 | • | Box.Fill.PEN | (Use PEN) |
| 98 | • | SCROLL UP | (Ring Scroll Up) |
| 99 | • | SCROLL DOWN | (Ring Scroll Down) |
| 100 | • | SCROLL LEFT | (Ring Scroll Left) |
| 101 | • | SCROLL RIGHT | (Ring Scroll Right) |
| 102 | • | Line | (Use PEN) |
| 103 | • | Circle | (Use PEN) |
| 104 | • | Circle.Fill | (Use PEN) |
| 105 | • | Polygon | (Polygon, frame only) |
| 106 | • | Polygon.Fill | (Polygon.Fill) |
| 107 | • | SUPER.Fill | (Scan Line data) |
| 108 | • | Screen.Effect | (User Screen effect) |
| 109 | • | PSET.PEN | (Point Set) |
| 110 | | DRAW | |
| 111 | | DRAW Jump Address | |
| | | -------------------- FONT MANAGER -------------------- | (2500H) |
| 116 | • | MESSAGE.LEFT | (Message Left -> Right) |
| 117 | • | MESSAGE.RIGHT | (Message Right -> Left) |
| 118 | • | MESSAGE.CENTER | (Message Centering) |
| 119 | • | PRINT.ONE.CHR | (1 character print) |
| 120 | • | MESSAGE | |
| 121 | • | MESSAGE.LENGTH | |
| 122 | • | SET.FONT.RECORD | (Font record set) |
| 123 | • | READ.FONT.RECORD | (font record read) |
| 124 | • | SET.XY | (X, Y set to current Record) |
| 125 | • | READ.XY | (Read X, Y current record) |
| 126 | • | CR0A | (Line Feed) |
| 127 | • | CR0D | (Return) |
| | | -------------------- PICTURE DRIVER II -------------------- | (2E00H) |
| | | -------------------- EVENT MANAGER -------------------- | (3000H) |
| 132 | • | GAME.KEY.REPEAT | (GAME.Key, Repeat process) |
| 133 | • | GET.CLICK.STATUS | (read Click Status) |
| 134 | • | KEY.REPEAT | (1 Key Repeat process) |
| 135 | • | SET.EVENT.RECORD | (Event record set) |
| 136 | • | READ.EVENT.RECORD | (Event record read) |
| 137 | • | SAVE.CURSOR | (KVRAM character save) |
| 138 | • | LOAD.CURSOR | (KVRAM character read) |
| 139 | • | FLASH.CURSOR | (Flash cursor) |
| 140 | • | MODIFY.KEY | (Modify Key Read) |
| 141 | • | SET.KEY.X | (Asc Table Set) |
| 142 | • | TEXT.EDIT.LINE | (Text Edit of 1 Line) |
| 143 | • | BEEP | (Bell) |
| 144 | • | MEMORY.COPY | (Memory Copy, Move) |
| 145 | • | MEMORY.SWAP | (Memory Swap) |
| | | -------------------- WINDOW MANAGER ---------------- | (3400H) |
| 148 | • | SET.WINDOW.RECORD | (Window record set) |
| 149 | • | READ.WINDOW.RECORD | (Window record read) |
| 150 | • | SET.WINDOW.RECORD.EXT | (Extend Window Record set) |
| 151 | • | READ.WINDOW.RECORD.EXT | (Extend Window Record read) |
| 152 | • | PRINT.WINDOW.ONE | (Print KVRAM, Window one) |
| 153 | • | PRINT.WINDOW.BUFFER | (Print KVRAM, Window buffer) |
| 154 | • | PRINT.BG | (Print Back Ground Pattern) |
| 155 | • | PRINT.SCREEN | (Print KVRAM -> V-RAM) |
| 156 | • | CHECK.XY.IN.SCRN | (Check xy in Screen) |
| 157 | • | CHECK.XY.IN.WINDOW | (Check xy in Window) |
| 158 | • | CHECK.CLICK.POINT.BF | (Check click point where) |
| 159 | • | ID.JUMP | (To Subroutine by Item.ID) |
| 160 | • | FUNCTIONS | (Low-level function process) |
| | | -------------------- Load and Save Routine ---------------- | (4600H) |
| 161 | • | LOAD&SAVE.MODULE | (File Save and Load) |
| | | -------------------- WINDOW MANAGER ---------------- | (4800H) |
| 162 | • | SERVICE ROUTINE | (Service call) |
| | | -------------------- SUB.FUNCTION NO. -------------------- | |
| 0 | • | QUIT | (Application End Call) |
| 1 | • | EVENT.LOOP | (System Event Loop) |
| 2 | • | ACTIVE.WINDOW | (Inactive to Active) |
| 3 | • | INACTIVE.WINDOW | (Active to Inactive) |
| 4 | • | WAIT.ON.EVENT | (Wait any Event) |
| 5 | • | MOVE.CURSOR | (Move Cursor in Screen) |
| 6 | • | WINDOW.XY+SCREEN.XY | (Window X, Y + Screen X, Y) |
| 7 | • | PRIORITY | (Priority Change) |
| 8 | • | FLASH.ITEM | (Flashing Select Item) |
| 9 | • | DRAG | (Drag by DRAG Record) |
| 10 | • | VIEW.CLIP.CHR | (CLIP.SCREEN.RECT) |
| 11 | • | VIEW.CLIP.ATB | (CLIP.SCREEN.RECT) |
| 12 | | | |
| 13 | • | NEW.WINDOW | (New Window Set) |
| 14 | • | CLOSE.WINDOW | (Window Close) |
| 15 | • | ALWAYS.WINDOW.ACTIVE | (Window Active) |
| 16 | • | SET.DUMMY | (System.ITEM.Dummy.Mode) |
| 17 | • | PLUS.ITEM.XY | |
| 18 | • | PLUS.nXY&WXY | |
| 19 | • | SUB.ITEM.XY | |
| 20 | • | SERCH.ITEM.TYPE | |
| 21 | • | SERCH.ITEM | |
| 22 | • | SERCH.DATA | |
| 23 | • | BIN.TO.DEC | |
| 24 | • | BIN.TO.HEX | |
| 25 | • | SERCH.DATA 2 | |

## インタフェイスガイドライン

Macintoshにはヒューマンインタフェイスガイドというのがあって，Macintoshのソフトはこのガイドラインを守って作られているのがほとんどです。SYSTEM-7Cではあまり，細かいことはいいませんが，次のことを守っていただけたら幸いです。

★なるべく英語は避けてください。

★アプリケーションには最低ひとつ（メイン）メニューを入れてください。

★メインメニューにはクローズボックス，リサイズボックスは付加しないでください。

★メインメニューの先頭項目は“～について”，最終項目は“Quit”としてください。

★簡単なソフトの説明を入れてください。通常“～について”が選択されたら表示されるようにしてください。

★アイテムが選択されたら，なるべく画面になんらかの変化を持たせてください。

SX-WINDOWにもいずれインタフェイスガイドラインが提示されると思いますが，たぶんMacintoshと同じようになるでしょう。だいたい，中身がMacintoshにそっくりみたいな感じですから。

## 注意事項

ソースリストは配布するので，自由に改造，改良してください。ただし，改造，改良した結果発生したトラブルに関しては，改造，改良した本人が責任を持ってください。なにかできたら，EXTRAとOh！X編集部へ送りましょう。

また，ゲームを作成しているとよくあることですがメモリが足りないことがあります。そのときはSYSTEM-7Cで使用しないルーチンを削除してください（ソースリスト上でもオブジェクト上でもどちらでも可)。

**●SYSTEM-7Cのゲームへの組み込み**

System-7Bではどちらかというと別々にロードしたりすることが多かったので面倒でした。そこでSYSTEM-7Cを使用したゲームはSYSTEM-7Cとゲームをまとめてセーブしてください。カスタム化したSYSTEM-7Cを組み込んでくれてもかまいません。ただし，その場合でも“SYSTEM-7C使用”と明記してください。

スペースグラフィックのPUTルーチンで，キャラクタ画面とカラー画面に書き込むためには，PUT.CHRとPUT.ATBなどのように対応するPUTルーチンを使用してください。

SYSTEM-7Cは階層構造になっているため，先月号の図2の上層にあるマネージャは下層マネージャがないと動作しません。つまりスペースグラフィックはカーネルがないと動作しませんし，ウィンドウマネージャはイベントマネージャがないと動作しません。注意してください。ただし，ピクチャードライバはフォントマネージャがなくても動作可能，ロードセーブモジュールは単独でも動作可能となっています。

SYSTEM-7Cはあらかじめ初期値が設定されています。そのためユーザーがいちいちプログラムで初期設定をする必要はほとんどありません。ただし，イベントレコードを除くレコード関係は初期化されていません。また，カーネルにあるASCIIコード入力はSystem.Configによってキーマップが設定されています。そのためカーネルだけでASCIIコード入力を行う場合はプログラム上で設定する必要があります。キーマップはユーザーがSystem.Configを変更することにより独自のキー配置にすることができます。

ビットマップフォントデータはなにも設定されていません。使用する場合は，プログラム側で設定する必要があります。

フォントマネージャはキャラクタ文字表示時，ROMモニタを使用します。そのため，マネージャを使用する場合は必ずバンクをROMにしておく必要があります。

割り込みはなるべく使用しないでください。ただし，ゲームで使用する場合はかまいません。

アプリケーションを作成するときはなるべくプログラムアドレスは高位アドレス（$A000_H$など）に作成してください。低位アドレスはSystem.Configによって使用されるおそれがあります。

SYSTEM-7CにはSX-WINDOWのShell，MacintoshのFinderに相当するものがありません。アイコンがないよーとわめかないように。

**●雑談**

ゲームを作るよりもウィンドウシステムを作るほうが楽でした。ゲームのほうが難しい。特に移植なんか大変です。最近Oh！X誌上でゲームのダンプリストが載ることが少なくなりました。SYSTEM-7Cも見てのとおりダンプリストがありません。SYSTEM-7C関係のものはすべてEXTRAで配布するからです。なかには残念だと思う人がいるかもしれませんね。ダンプリストを入力するのもたまにはいいと思います。S-OSではしっかりダンプリストが載っていますけど。

いままで，SYSTEM-7Cのウィンドウマネージャのスピードについては，あまり触れませんでした。なかには怪しげに考える人がいるでしょうから，最後のほうになったけど，書くことにしましょうか。結論からいえば，速い！　ちなみにこの速さは余裕の速さです。つまり，もっと速くできます。で，どのくらい速いかというと，SX-WINDOW（旧バージョン）の10倍は速いかな（見た目ですよ)。MacintoshIIくらい。

まあ，ウィンドウの大きさや枚数にもよりますが，ドラッグしてボタンを離すともう，すべて書き換えが終わっているくらいです。最初ウィンドウシステムのウィンドウ表示ルーチンを作って表示させたとき，相当のスピードで描画してくれたので，あとあとスピードを落とすようにプログラムしてあります（あまり速いと，ほかのウィンドウシステムに悪いから)。が，それでもスピードの遅いところがありました，カセット(笑)。FDDつければ仮想記憶なんてできるのにね。でも，うちのMZ-700はカセットでグリーンディスプレイ（一応カラーCRTもあるけど）でプリンタなしだから，駄目ですね。他人から見れば，この開発環境は最低に見えるでしょう。本人もそう思います。でも，ディスクがついてもひとつのプログラムを仕上げるスピードは変わらないような気もします。思考がそんなに速くないので，MZ-700でもなんとかやっていけるのです。Macintoshも遅いので試行錯誤には結構いいかな。

## 終わりに

MZ-700でよくここまでやってきたなって感じです。MZもX68000もMacintoshもNeXTもコンピュータであることに変わりはありません。NeXTで動けばなんとかMZでも動く可能性があるわけです。私がウィンドウシステムを作ったからといってゲームのことを忘れたわけではありません。むしろ，ゲームをより早く手軽に簡単に作るためにウィンドウシステムを作ったのです。Macintoshに触れるようになって，もっと手軽にゲーム開発ができないかなと考えるようになりました。SYSTEM-7Cはそういう考えのもとに作られたのです。

最後にEXTRAの会長である筑紫さん，会社（Family Video Systems）の藤原さん，高橋さん，その他いままで応援してくれた方々に深く感謝いたします。それでは，また誌面でお目にかかりましょう。

# CARD PRO-68K の応用

Ogikubo Kei 荻窪 圭

今回は「CARD PRO-68K ver.2.0」の使い方，つまりデータ入力やマクロの作り方などを紹介しています。それとともにもう少し本格的にアンケートを分析したかったのですが，作者の都合により，単なるサンプル使用に留まる形になっています。

「数字の客観性などというのは幻想である。言葉と同じく，数字による表現も人間の抽象化の産物である。それは送り手と受け手の双方によってさまざまな解釈を下される。数字は本来的なパラドックス，混乱，矛盾，そして隠された問題点によって，別の言語を構成しているにすぎない」(ウィルソン・ブライアン・キイ著「メディア・レイプ」より)
ってなわけで，予告どおり，「大人のためのX68000 第10回」は始まる。

## データを入力する

今回はちゃんとデータを入力するわけだが，「CARD PRO-68K」に向かって1つひとつ入力するも，エディタでまとめて入力しておいて読み込ませるもどっちでもいい。私は後者を採用した。

できるだけ楽に入力したいからエディタを使った。すると，かな漢字変換がうっとうしい。さらに，X68000の型番をいちいち入れるのもうっとうしい。だから，○を0に，×を1にするとか，男を“<”に，女を“>”にするといった工夫をする。

そうやって入力した様子が写真1だ。データは全部で100件。一応無作為抽出。セパレータはカンマ。これをエディタの置換を使ったり，「CARD PRO-68K」に読み込んでからマクロを使って書き換えるなどして，写真4になるわけである。へへへ。

データベースにデータを入力するとき，一般論として以下のことに注意するべし。それは，英数字を全角にするか半角にするかに統一することと，長音とハイフンと漢数字の一など，よく似た文字を間違えないよう注意することである。見た目はよくても，検索時にひっかかるからだ。特に，半角のハイフンと長音は区別がつきにくいので，住所録や顧客名簿では（特に，入力を人に頼むときには）よりいっそうの徹底が必要となる。

さて，このテキストデータを「CARD PRO-68K」へコンバートする。「CARD PRO-68K」はK3式と呼ばれるCSVをサポートしている。これは，文字列だけを「"」で囲んだCSVだ。しかし，表1の「"」なしのファイルでも問題なく，よきにはからってくれた。ラッキー。

コンバートはユーティリティメニューにある。写真1が実行例だ。すると，自動的に項目長や項目タイプ，項目数を設定してくれるので，簡単にデータベースファイルが作れてしまう。気持ちがいい。

コンバートが終了したら（入力ミスの発覚などがあって，けっこう時間がかかったが），項目名の設定や見直しを行う。

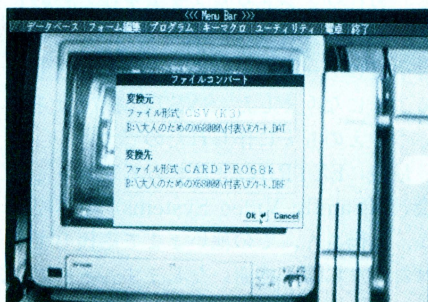
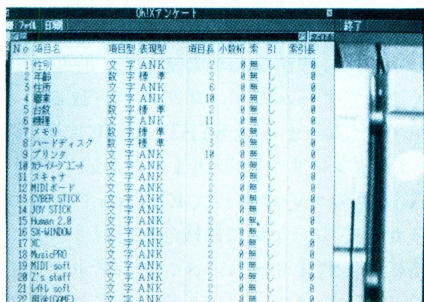
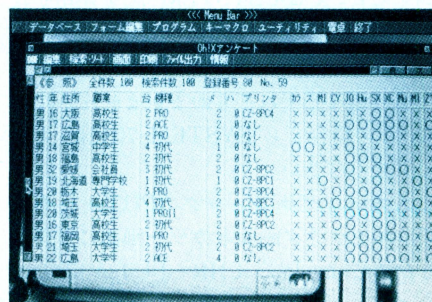
これにはデータベースコマンドのデータベースの定義・作成を使う。写真2である。このようにして項目名をセットしていく。全部で33項目。

## データの参照

たった100件ではあるが，データが集まった。とりあえず眺めてみよう。データベースコマンドの，“操作”だ。“操作”を選んで，操作したいデータベースを選択すればいい。すると，標準画面になる。1項目1行で表示されるものだ。

これではつまらないので，全体を眺めたいときの一覧表にする。画面メニューだ(写

**表1**

```
<, 32, 愛媛, 会社員, 3, 初代, 2, 0, CZ-8PC2, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, @2, @2, @2, 荻窪圭
<, 19, 東京, 浪人, 1, 初代, 2, 0, なし, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, @3, @1, @1, なし
<, 21, 静岡, 大学生, 2, 初代, 2, 60, VP-800, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, @2, @3, @3, 荻窒圭
<, 17, 千葉, 高校生, 1, 初代, 1, 0, CZ-8PC2, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, @3, @0, @3, 荻窪圭
<, 35, 茨城, 公務員, 1, 初代, 2, 40, CZ-8PK, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, @3, @2, @1, なし
```

❶CSVからコンバート

❷データベースの項目定義

❸コンバートしたデータベース

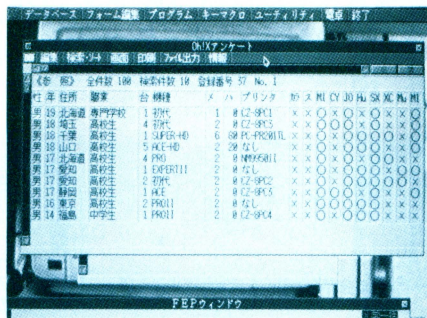
真3)。項目間やレコード間の区切りが罫線ではなく，陰影だっていうところがミソ。3Dな雰囲気が高級感を醸し出す。この場合，項目数が多いので，画面に収まらない。んなときはどうするかというと，一覧表画面をカスタマイズするのである。フォーム作成メニューから一覧表メニューを選ぶ。

フォーム作成は，「CARD PRO-68K」を極めるポイントのひとつだ。データベース操作ウィンドウからあらかじめフォーム編集メニューで作った画面を使うことができるのだ。逆にいえば，データベース操作ウィンドウで画面を作ることはできない。そういうときは，メインのメニューバーからフォーム編集を選ぶだけだから，面倒なことはないけれど。また，ひとつのデータベースにいくつものフォームが設定できるので，いろんな視点の表が得られる。捨てがたい魅力だ。

とはいえ，一覧表画面には欠点がある。写真3を見てもらえばわかるとおり，表示項目長はデータベース定義で定義した項目長に支配されるため，項目名が切れてしまう。これは困ったものだ。自動的に項目名を複数行表示してくれるとかするとうれしい。項目長を長くすればいいのだが，そうすると，1画面に映る情報量が激減するしね。

私は一覧表画面が好きだが，業務なんかで使う場合はそうもいかない。1つひとつのカードに意味を持たせたいときはそれなりにカスタマイズされた画面がうれしい。そういうときは，フォーム編集で自由画面を作る。今回は特に作らなかった。

## クエリー

クエリーってのはQUERYのことで，“質問する”という意味だ。いや，ただカタカナで書くと妙な感じで面白いから見出しにしただけ。データベースの世界では検索条件を指定することをQUERYということが多い。質問事項を作るって意味だろう。んで，検索はサーチだ。

❹検索条件入力ダイアログ

❺10代でMIDIボードを持っている人

さて，写真3のデータベースがある。一覧表だけを眺めていてもしょうがないから，データベースソフトの基本である，“検索”をしてみる。「CARD PRO-68K」の検索メニューは“抽出”といったほうが通りがいい。“検索”というと，エディタやワープロの検索機能のようなものを想像しかねないからね。というのは，検索を行うと，ウィンドウ内に表示されるデータが検索条件にあったものだけになるからだ。

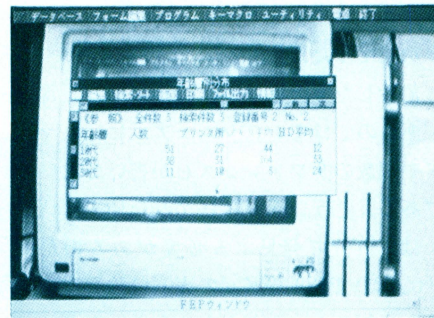
検索のメニューが写真4である。このようなダイアログが開き，検索条件を指定する。写真4は，年齢が10代で，MIDIボードを持っている人，って検索条件だ。マウスでスススイである。あいまい検索，ってのはできないが，別に気にならない。

とりあえず，ポイントは検索条件のロード/セーブが可能なことだ。定型な業務には欠かせない機能である。

検索結果が写真5。100人中10人が条件に合致している。10代の少年は（あ，ちなみに，100人中女の子は0だったから少年でいい）全部で51人だったから，約5分の1が対象だ。

あまり面白くなかったかな。

なお，写真5は年齢順でソートをかけてある。ソートの要領も検索と同じだ。ロード/セーブも可能。

抽出されたデータに対してさらに検索しようとすると，“絞り込み検索モード”となる。ANDを指定すると，抽出データに対して検索がかかり，ORにすると，全データに対して，現在の検索結果とのORがとられる。再び全データに対して新たに検索を始めたいときは“全検索”を選べばいい。

❻マクロを使って自動作成したもの

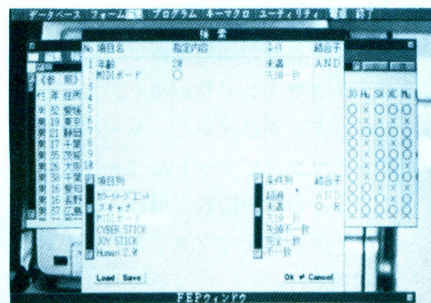
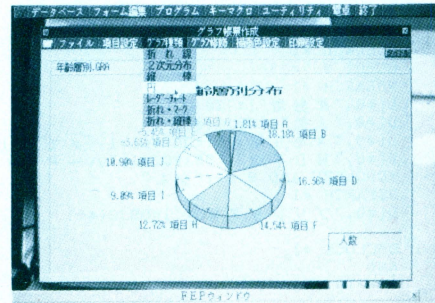
## マクロを作ろう　ーその1ー

ここで集計マクロに挑戦する。マクロでも使わないと，データ分析ができないからだ。マクロってのは，「CARD PRO-68K」でいうプログラム機能のことね。

ここで挑戦するマクロは“元のデータベースの集計を，別のデータベースに書き込む”ものである。具体的には，元のデータを10代，20代，30代の3つに分け，それぞれで人数，プリンタ所有者数，平均メモリ，平均ハードディスク容量を算出し，書き込むことにした。もっといろいろやりたかったが，マクロを組むのはけっこう時間を食うのだ。

先に結果を見せてしまうと，写真6である。平均メモリ容量がヘンな値になっているが，原因は不明である。元のデータに変なのが混じっていたのかもしれない。うーん。考えておこう。

で，あらかじめ，データベース定義だけしておいてリスト1のマクロを実行すると，自動的に写真6ができあがるのだ。

いいねえ。「CARD PRO-68K」の醍醐味ってやつだねえ。もっとも，このBASICのような強力なマクロを完成させるのは大変だった。あまりにも大変だったので，サブルーチンも使わず，無駄なプログラムになってしまった。だって，サブルーチンにす

❼グラフ帳票を設定する

るより，カット&ペーストで増殖させたほうが楽だったんだもん。

苦労した原因はマニュアルにある。言語が悪いのではない。マニュアルにはコマンドや関数のリファレンスしか書いてないのである。どういうときにどれを使えばいいかは，自分で発見する必要があるのだ。「CARD PRO-68K」の売りは強力なマクロなのだから，そこんとこをちゃんとマニュアルにしてほしかった。これではBASICや簡易言語の経験がないと苦労するだろう。

で，簡単な解説は，リストに入れておいた。ポイントは，使うデータベースを最初にオープンする必要があることと，APPENDを入れると，データ追加モードになることだ。

## グラフを描いてみる

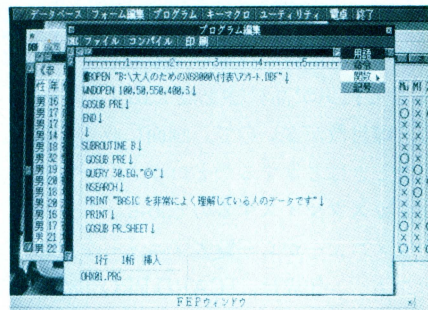
いよいよグラフである。グラフは，写真6の年齢層別分布データから，年齢層別円グラフを作るのだ。

これにはまず，フォーム編集のグラフ帳票を選択する。そして，その帳票を使うデータベースを指定すると，グラフウィンドウが現れる。

ここで，グラフ化する数値項目と，縦軸に使う名前をデータベースの項目から選ぶわけだ。すると，“実際のデータが入っていないダミーのグラフ”が表示される（写真7)。こいつをセーブする。

あとは年齢層別データベースを開いて，画面メニューからグラフを選択するだけだ。そうすると，データベースの表の代わりに，グラフがべろんと表示される。

このグラフ描画機能の秀逸な点をいくつか挙げておこう。

❽帳票にデータを当てはめる

❾マクロ作成画面

まず，ウィンドウの大きさに応じて，グラフの大きさだけでなく，文字の大きさも変わること。

続いて，データを抽出した状態でグラフ描画すると，抽出データを対象にグラフ化してくれること。いろいろ条件を変えたグラフが見られて便利である。

最後に，ひとつのデータベースに対し，グラフ帳票をいくつも（といっても制限はあるが）設定できること。まさしく，視点を変えてデータを眺められる。

写真8は，写真7で作成したグラフに実際にデータをあてがったところだ。なるほど，10代が全体の半分を占め，続いて20代，いちばん少ないのが30代だというのがわかる。40代の人は無作為抽出の100件にはいなかった。30代の人は意外と多かった，というのが私の感想だ。

なお，ここでは円グラフにしたが，2次元分布や，縦棒，折れ線，レーダーチャートなども選べる。

グラフ専用ソフトではないため，縦軸の最大値を変更したり，グラフのレイアウトを変えたりはできない。そんなもんだ。

## マクロを作ろう　ーその2ー

マクロをまたもや作る。ここではもっとインタラクティブなやつだ。

作成手順を少し詳しく示そう。

写真9がマクロの編集画面だ。このように，エディタウィンドウが開く。実のところ，このエディタは遅いし，キーバッファがたまる。マウスでカット&ペーストができることと，命令や関数，項目名やデータベースファイル名をマウスで選択できることだけが救いだ。

が，そんなのはいやなので，雛型をこの

**リスト1**

```
REM Oh!X 読者年齢層別分布                                   → 最大DB数
FILES 2
DBOPEN "B:¥大人のためのX68000¥付表¥ｱﾝｹｰﾄ.DBF" AS #1     } DBのオープン
DBOPEN "B:¥大人のためのX68000¥付表¥年齢層別.DBF"           } #1，#0，(省略時)
REM 10代
APPEND                                                      → 書き込みモード
FINDALL #1
QUERY 2,<,20                                                } 全体から20歳未満の検索
NSEARCH #1
D[#0,"年齢層"]="10代"
D[#0,"人数"]=OBJC(#1)                                       } 人数，平均値を書き込む
D[#0,"メモリ平均"]=DBAVG(#1,"メモリ")
D[#0,"HD平均"]=DBAVG(#1,"ハードディスク")
ADDQUERY AND,9,<>,"なし"
NSEARCH #1                                                  } プリンタ所有者を検索
D[#0,"プリンタ所有者"]=OBJC(#1)

REM 20代
APPEND
FINDALL #1
QUERY 2,<,30
ADDQUERY AND,2,>=,20
NSEARCH #1
D[#0,"年齢層"]="20代"                                       } 上の20代版
D[#0,"人数"]=OBJC(#1)
D[#0,"メモリ平均"]=DBAVG(#1,"メモリ")
D[#0,"HD平均"]=DBAVG(#1,"ハードディスク")
ADDQUERY AND,9,<>,"なし"
NSEARCH #1
D[#0,"プリンタ所有者"]=OBJC(#1)

REM 30代
APPEND
FINDALL #1
QUERY 2,>=,30
NSEARCH #1
D[#0,"年齢層"]="30代"
D[#0,"人数"]=OBJC(#1)                                       } 同じく30代版
D[#0,"メモリ平均"]=DBAVG(#1,"メモリ")
D[#0,"HD平均"]=DBAVG(#1,"ハードディスク")
ADDQUERY AND,9,<>,"なし"
NSEARCH #1
D[#0,"プリンタ所有者"]=OBJC(#1)

DBCLOSE #1                                                  → DBのクローズ
END
```

❿マクロで作ったウィンドウ

⓫オマケ

エディタで書いたらセーブし，チャイルドプロセスでED.Xを使ってエディット。

なお,「CARD PRO-68K」の専用エディタではセーブする際に自動的にマクロがコンパイルされる。ここでエラーチェックがなされるので，結構うれしい（もっとも，ここでエラーになんなくても，実行してみたらエラー，ってなケースは頻繁に生じる。当たり前といえば当たり前だ)。

できたのがリスト２である。実行したのが写真10の画面である。

何が起きているか。実行結果表示用のウィンドウを開き，そこにボタンをつける。そんでもって，ボタンをクリックするとそれに応じた計算がなされて，結果がウィンドウ内に表示されるというカッコいいマクロだ。

リストの説明をちゃんとしておこう。けっこう,「CARD PRO-68K」のエッセンスが詰まっているからだ。

まず，リストにコメントしてあるように，このプログラムはメインルーチンとサブルーチンとイベント処理からなっている。イベント処理というのは，SX-WINDOWのイベントと語源は一緒である。「CARD PRO-68K」のマクロでは，いくつかのイベントが設定されており，プログラム実行中にそのイベントが起こったとき，イベント処理ルーチンが割り込みで実行されるのだ。うーん，おしゃれ。

リスト２では，ウィンドウ内のボタンが押されたらイベント処理が起きるように書いてある。イベントには，データベースがオープンされたら，だとか，データ入力がなされたら，などなど６種類。

“データ入力があったイベント”は，自動計算のときに便利だ。たとえば，ある項目とある項目の計算結果を別の項目に埋め込みたいときは，イベント処理を使う。指定した項目にデータが入ったら自動的に計算して，結果を別の項目に入れる，なんていうマクロも書けるのだ。

ここではウィンドウにボタンを３つ設定しているので（ボタンに名前はつけられなかった),ボタンが押されたイベントをチェックし，そのあとボタン番号で処理を分岐している。

サブルーチンは５つあり，２つが汎用の計算やデータ表示，３つがBASIC, C, アセ

**リスト2**

```
DBOPEN "B:¥大人のためのX68000¥付表¥ｱﾝｹｰﾄ.DBF"     } メインルーチン
WNDOPEN 100,50,550,400,3                        } ウィンドウオープン
GOSUB PRE                  └→ボタンの数
END

SUBROUTINE B                                     } サブルーチン
 GOSUB PRE
 QUERY 30,EQ,"◎"
 NSEARCH
 PRINT "BASIC を非常によく理解している人のデータです"
 PRINT
 GOSUB PR_SHEET
RETURN

SUBROUTINE C
 GOSUB PRE
 QUERY 31,EQ,"◎"
 NSEARCH
 PRINT "  C   を非常によく理解している人のデータです"
 PRINT
 GOSUB PR_SHEET
RETURN

SUBROUTINE A
 GOSUB PRE
 QUERY 32,EQ,"◎"
 NSEARCH
 PRINT "ｱｾﾝﾌﾞﾗを非常によく理解している人のデータです"
 PRINT
 GOSUB PR_SHEET
RETURN


SUBROUTINE PR_SHEET
   PRINT
   PRINT "  平均年齢                  ",DBAVG("年齢")," 歳"
   PRINT "  平均パソコン所有台数      ",DBAVG("台数")," 台"
   PRINT "  平均メインメモリ          ",DBAVG("メモリ")," MB"
   PRINT "  平均ハードディスク        ",DBAVG("ハードディスク")," MB"
RETURN                                └→平均

SUBROUTINE PRE
 WNDCLEAR
 PRINT "ボタンをクリックしてください。"
 PRINT "１がBASIC、２がC、３がアセンブラに◎をつけた人の各種データです"
 PRINT "まず、全員対象ののデータです"
 FINDALL
 GOSUB PR_SHEET
 PRINT
 PRINT
RETURN

EVENT BUTTON                  } イベント処理
  N=BUTTONNUMBER              } BUTTONNUMBERは
  SWITCH N                    } 押されたボタンの番号
   CASE 1:
     GOSUB B
     EXITSW
   CASE 2:
     GOSUB C
     EXITSW
   CASE 3:
     GOSUB A
     EXITSW
  ENDSW
ENDE
```

ンブラのそれぞれの処理を受け持っている。簡単だね。

なお，ウィンドウ内のボタンの数は，WNDOPENコマンドの最後のパラメータで指定している。

そのほかのポイントとしては，FINDALL コマンドがデータベース操作ウィンドウでいう“全検索”，“QUERY”コマンドは検索条件指定，“NSEARCH”コマンドは“QUERY”コマンドで指定した条件での検索実行である。また，PRINT文にあるDBAVG関数は，指定した項目の平均を算出する関数だ（データベースアベレージね）。ちなみに，「CARD PRO-68K」は62個の命令と58個の関数を誇っている。

## 言語と年齢の関係

写真11は，“C言語をよく理解している”と答えた人と，全体の平均を比べたシーンだ。

全体のメモリの平均はまたもやぶっとんでいるが，Cのほうはそうでもないから信用してもいいだろう。

Cプログラミングユーザーの特徴は，年齢層が高い。これにつきる。また，パソコンを複数台所有しているってのもいえるな。ちなみに，BASICユーザーの場合は21.1歳，1.4台，2.16Mバイト，18.1Mバイト。アセンブラユーザーの場合は21.6歳，2.06台，2.56Mバイト，30Mバイトとなった。

とりあえず，どれかのプログラミング言語に◎をつけた読者の平均年齢は全体の平均年齢より高いことがわかる。また，BASICユーザーにハードディスク持ちは少ないこともいえるかもしれない。

## マクロを作ろう　－その3－

さて，いままで作ったマクロは，マクロを実行すると，そのマクロの環境で勝手にマクロが動き，せっかくオープンしたデータベースはただの飾りというものだった。

今度は違う。リスト3だ。これはただ単に，アンケートの単純集計をウィンドウに表示するだけのマクロ。もはやお馴染みの，年齢と所有台数と平均メモリと平均ハードディスク容量を算出するもの。ポイントは，FINDALLコマンドがないことだ。

使い方であるが，実行するとアンケートのデータベースがオープンされ，その上に，ボタンがひとつあるウィンドウ，プログラムウィンドウが重なる。そしたら，データベースウィンドウをクリックしてそれをアクティブにし，集計の元を作るのだ。検索・抽出を行うのである。

そして，プログラムウィンドウに戻り，ボタンをクリックすると，抽出した結果に対して計算がされるのである。

楽しいではないか。

で，このマクロと検索機能を使って，先月判明した人気ライター上位3人それぞれに投票した読者と，実はいちばん多かった“なし”と答えた読者。それぞれの平均年齢を求めてみた。

ここに書いてしまうと，

| | |
|---|---|
| 全体の平均 | ：20.9歳 |
| 西川善司 | ：18.5歳 |
| わたし | ：19.9歳 |
| 祝一平 | ：21.4歳 |
| なし | ：21.5歳 |

である。

西川善司氏の支持者はやはり10代であった(おいおい選挙じゃないって)。わたしは20歳くらいはいくと思ったのだが，全体の平均をひとつ下回ってしまった。祝一平氏ももっと上になると思ったのだが，“なし”の人と同じくらいになった。つまるところ，歳をとると，こういう雑誌にライターの個性を期待するということがなくなってくるのだと思う。だから，“なし”と答えた人の平均年齢が高いのだ。おそらく。

## CARD PRO-68K ver.2.0の総評

まだ印刷機能の話をしていなかったが，今回は割愛だ。残りページも少ないので，ここで締めに入る。

「CARD PRO-68K」を使っていて感じるのは，ユーザーインタフェイス。妙なスクロールバーや，カット＆ペーストの操作性（特に，どこででも使えるわけではないこと）などなど，Macintoshなんかと比べるとまだまだ洗練されていず，使いにくい面がある。が，MS-DOSのカード型データベースに比べればずっといい。

あとは先月に述べた柔軟性だけだ。

「CARD PRO-68K」が売りにしている強力なマクロは，今回自分で組んでみてなかなか遊べることがわかった。前バージョンとは格段の違いだ。マニュアルさえよければ，もっと使ってもらえるだろう。専用のエディタもいまいちだけどね。

価格も，内容を考えたらとてもよいコストパフォーマンスの29,800円。プログラム機能を使う人にとっては安いもの，と断言しよう。

というわけで，前バージョンとは大きく進歩した「CARD PRO-68K」。そういうことだ。定型業務にも使えるというのは強みだろう。

来月は，製品版が間に合ったら，の話だけど，Multiwordでもやりますか。希望も多いようだし。Multiwordをを題材に，正しいレイアウトワープロのありかたを考えてみるのもいいかもしれない。

あ，今回は「CARD PRO-68K」の使い方の紹介のほうに重点を置いていて，データは100人だけのサンプル使用だから，結果はどこまで信用していいかはわからないよ。では，おやすみなさい。

**リスト3**

```
DBOPEN "B:¥大人のためのX68000¥付表¥アンケート.DBF"      } メインルーチン
WNDOPEN 100,50,400,200,1
END

SUBROUTINE PR_SHEET                                    } サブルーチン
   PRINT
   PRINT "アンケートの単純集計表"
   PRINT
   PRINT "平均年齢                    ",DBAVG("年齢")
   PRINT
   PRINT "平均パソコン所有台数        ",DBAVG("台数")
   PRINT
   PRINT "平均メインメモリ            ",DBAVG("メモリ")
   PRINT
   PRINT "平均ハードディスク          ",DBAVG("ハードディスク")
RETURN

EVENT BUTTON                                           } イベント
 GOSUB PR_SHEET
ENDE
```

## 吾輩はX68000である
## ［第3回］

# 我が好敵手C言語

Izumi Daisuke　泉　大介

近頃巷ではC言語が大流行ときく
なにゆえそれほどもてはやされるのか
その理由をみてみなければなるまい

世の中右を向いても左を向いてもC言語ばかり。なんでも書店の棚はC言語の解説本によって溢れかえり，占領されんばかりの勢いだそうだ。かつて，PASCALという言語がBASICのあとを受け継ぐ言語として注目されたことがあったが，結局ひと握りのマニアを残すだけで今日では教育目的以外には見向きもされない言語になっているのと同様，C言語も一時の流行りで終わるかと思いきや，さにあらず。いまやゲームプログラムまでC言語で書かれるようになり，プログラミングの中心言語として定着したようである。

御仁はかつてはZ80のマシン語でならしたこともあり，Z80より遙かに洗練された命令を持っているMC68000を搭載した吾輩がやってきたからには，心ゆくまで，思う存分，マシン語ライフを満喫し，果てはとんでもないプログラムを作ってもらえるものと期待していたのだが，吾輩がやってきて4年が過ぎたいまになっても，まだ一度もアセンブラを使おうとはしてくれない。Cコンパイラのβバージョンが編集室に届くまではグラディウスで遊び呆け，届いてからというもの今日までC言語ばかり使っている。質実剛健で有名なOh!Xでも，掲載されるプログラムの多くがC言語で書かれるようになってきており，アセンブラ，ひいてはマシン語でプログラムを作るという行為は過去の遺物になりつつあるのか，と一抹の不安を禁じ得ない。

## 敵を知り，己を知れば……

マシン語プログラム師であった御仁の進むべき道を違えてしまったC言語とはいかなる言語なのか。なぜ御仁はマシン語よりC言語を選んでしまったのか。御仁をまっとうなマシン語の道へ戻す第一歩は，C言語がなぜこれほどまでに使われるのかを探ってみることから始めなければなるまい。

### ●汎用性を考える

かつてはUNIXの動くワークステーションやミニコン上でしか利用できなかったC言語だが，いまでは数多くのパソコンに移植され，ほとんどすべてのマシン上でそれぞれのC言語が利用できるようになっている。これは，C言語さえ知っていれば，どんなマシンの前に座らされてもプログラムを作れることを意味する。ひるがえってマシン語は，同じMC68000をCPUに採用しているマシンでしか使うことができず，さらにMC68000を採用しているマシンであってもマシンが異なれば，あるいはOSが異なればそのままでは使い物にならないという弱点を持っている。

同じMC68000を採用しているのにマシンやOSが異なれば使い物にならない，という事実は，読者諸兄を少々混乱させるかもしれない。確かに，アドレスE00000$_{H}$に入っている1バイトデータをD0レジスタにコピーするという命令，

```
move $e00000,d0
```

は，どんなマシンでも，どんなOSを使っていようとも同じように動作する。確かに動作は同じなのだが，それに付随する意味を考えると，使い物にならないという理由がおわかりいただけるかと思う。X68000ではこの命令はテキスト画面の左上8ドットをD0レジスタにコピーするという意味を持っている。しかし，別のマシンではE00000$_{H}$番地にはメモリがないかもしれない。少なくとも，ここにテキストVRAMがあるという保証はまったくない。また，前回画面に「A」という文字を表示するマシン語のプログラムを紹介したが，このプログラムではIOCSを利用してフォントの格納アドレスを得ている。つまり，このプログラムは，吾輩と同じIOCSを持っているマシンでなければ動作しないことになる。これらの条件が異なれば，当然プログラムは違う形になるのである。

C言語（とライブラリ）が，

```
printf( "A" );
```

とやるだけで，マシンを問わず「A」の文字を表示できるのとは大きな違いだといえよう。

### ●計算の簡便さ

数値計算の容易さは，高級言語と呼ばれるものの専売特許といってもいいだろう。吾輩の扱えるアドレス上限はFFFFFF$_{H}$であることを先月お話しした。1アドレスには1バイトのデータを格納できるので，吾輩が扱えるメモリ量は000000$_{H}$～FFFFFF$_{H}$に入れることのできる1000000$_{H}$バイトである。これが何Mバイトになるかは，先月も触れたように，

(0xFFFFFF+1)/0x100000

で計算することができる。C言語で書くならば，

(0xは16進数を表す)/0x100000

である（0xは16進数を表す）。これをマシン語で計算すると，

move.l #$ffffff,d0　　FFFFFF$_H$をD0にセット

addi.l #1,d0　　それに１を加える

divu　#$100000,d0　そして100000$_H$で割る

となる，ところだが，実際にはそうはいかない。MC68000には，「ロングワード÷ロングワード」を計算する命令はないのである。最後の「divu」は「divu.w」の略記で，これは「ロングワード÷ワード」の計算を行う命令となる。つまり上のプログラムの最後の行は，100000$_H$のワード部分である0000$_H$でD0を割る，すなわちD0を0で割るという計算をすることになる。「ロングワード÷ロングワード」の計算を行うプログラムを用意しなければ，吾輩が何Mバイトのメモリを持っているかは計算できない（計算するまでのないという話もあるが）。

さらに困るのは実数の計算である。MC68000は実数をまったく扱うことができない。つまり，閏年を考慮して正確に１年が何日かを知ることはできないのである。実数演算は，数値演算プロセッサという専用のLSIを利用することになる。とはいっても，X68000では数値演算プロセッサはオプションなので，吾輩はその機能を肩代わりする計算プログラム集を持っている。実数演算にはこれを利用してもいい。図１がそのプログラムである。中に「__ltod」などと書いた行があるが，これが実数計算のプログラム集を利用している部分である。実数演算プログラム集の中には「ロングワード÷ロングワード」の計算プログラムも収められている。しかし，こうまでして１年が何日か知りたいと思うだろうか。しかも答えを画面に表示するには，さらにA982A$_H$番地から，

pea　$A9900

_print

というプログラムを書き加えなければならない。もちろん，「__ltod」は吾輩が持っている実数演算のプログラム集を利用しているので，これと同様のプログラム集を持っているマシンでしかこのプログラムは利用できない。「_print」のほうはHuman68kのシステムコールと呼ばれているものである。これもHuman68kが動いていなければ利用することはできない。

転じてC言語では，

```
void  main(void)
    {
      printf ("%g", (100-4+1)/400.0);
    }
```

というファイルを作ってコンパイル，実行するだけで，

**図１　マシン語での実数計算**

閏年の規則
　1）西暦が４で割り切れれば閏年である
　2）ただし，100で割り切れるなら閏年ではない
　3）ただし，400で割り切れるなら閏年である
よって400年間に，
　1'）100回閏年がある
　2'）４回閏年がない
　3'）１回閏年がある
以上より400年間の日数は365×400＋100－4＋1日
したがって１年あたりの閏日は，
　（100－4＋1）÷400
で計算できる

```
X68k Debugger v2.10 Copyright 1987,88,89 SHARP/Hudson
-p                                      ← プログラムを作っていいアドレスを確認
debug program from $0007BDB0
user  program from $000A9790           ← 示されるアドレスは，マシンの使用状況に
                                          よって異なる
-a a9800                                ← ここではA9800Hからプログラムを作成する
  000A9800      ori.b    #$00,D0
                move.l   #400,d0        ← 除数400をD0にセット
  000A9806      ori.b    #$00,D0
                __ltod                  ← それを実数に変換。答えはD0・D1に
  000A9808      ori.b    #$00,D0
                move.l   d0,d2          ← D0をD2にコピー
  000A980A      ori.b    #$00,D0
                move.l   d1,d3          ← D1をD3にコピー
  000A980C      ori.b    #$00,D0
                move.l   #100,d0        ← 被除数の計算。D0＝100
  000A9812      ori.b    #$00,D0
                subi.l   #4,d0          ← D0＝100－4
  000A9818      ori.b    #$00,D0
                addi.l   #1,d0          ← D0＝100－4＋1
  000A981E      ori.b    #$00,D0
                __ltod                  ← D0を実数に変換
  000A9820      ori.b    #$00,D0
                __ddiv                  ← D0・D1÷D2・D3を計算。答えはD0・D1に
  000A9822      ori.b    #$00,D0
                movea.l  #$a9900,a0     ← 文字列を格納するアドレスを設定
  000A9828      ori.b    #$00,D0
                __dtos                  ← 実数を文字列に直してA0に格納
  000A982A      ori.b    #$00,D0
                                        ← プログラム作成を中断
-b0 a982a                               ← ブレークポイントを設定
-g=a9800                                ← プログラムを実行

        break at 000A982A
PC=000A982A USP=00087ED4 SSP=000067F2 SR=0004 X:0  N:0  Z:1  V:0  C:0
D  00000032 70A3D70A 0000000E 00000000  00000000 00000000 00000000 00000000
A  000A9906 00000000 00000000 00000000  00000000 00000000 00000000 00087ED4
ori.b    #$00,D0
-ds a9900 a990f                         ← 格納された文字列を覗いてみる
000A9900  30 2E 32 34 32 35 00 30 30 30 30 30 30 30 30 30   0.2425.000000000
                                                              ↑これが答え
-q                                      ← デバッガの終了
```

計算を行い結果を画面に表示してくれる。なんと簡単！除数である400が400.0になっているのは，計算を実数で行うためである。これが400だと計算が整数で行われてしまうので答えは0になってしまう。整数の計算のほうが実数の計算よりも高速に行えるので，C言語は必要のない限り（つまり式の中に実数が現れない限り）計算を整数で行おうとする。スピードまでもが自動的に考慮されるわけである。ただ，8ビットマシン用のCコンパイラのなかには実数を扱えないものもある。

**●アドレスを直接操作する**

マシン語の専売特許といわれてきたハードウェアよりの処理に話を移そう。高級言語は，現在のメモリの状況がどうなっているのか，などという低レベルの（ハードウェアよりの）情報を，極力ユーザーに見せないようになっている。また，そうあるべきである。先の閏日の計算ではないが，目的は計算結果を得ることであって，それが図1のようなプログラムによって計算されていようが，もっとエレガントな方法で計算されていようが，そんなことは問題ではない。ましてやメモリの何番地にどんなデータが入っているかなどという生の情報は，ユーザーから隠されて然るべきである。逆にいえば，この目隠しされた状態を快く思わず，高級言語よりマシン語を愛好する人がいるのは事実といえよう。御仁もそうであった。

高級言語ではレジスタやメモリの代わりに変数を使う。整数を格納するための変数ｉが宣言されれば，整数を格納するためのメモリ（4バイト）が確保されて，以後ユーザーは「変数ｉにデータを格納」したり，「変数ｉに入っているデータ」を式の中で使ったりできるようになる。「変数ｉ」が実際にメモリのどこに確保されたかという情報も，「変数ｉ」にデータを格納するとこの確保されたメモリにデータが入れられるのだということも，あるいは式の中で「変数ｉ」を使うとこのメモリからデータが取り出されて使われるのだということも，ユーザーは知らなくていい。あたかも「ｉ」と名づけられた整数を入れるための箱があり，その箱にデータを入れたり，その箱からデータを取り出したりするような感覚で作業できるのである。

ところがC言語は，マシン語の専売であったはずの変数が確保されたメモリのアドレスという情報までユーザーに公開してしまった。変数の名前の前に‘&’をつければ，確保されたアドレスを知ることができる。また逆に，指定したアドレスにデータを直接書き込むこともできるようになっている。デバッガを使ってテキストVRAMに直接データをセットしたのは諸兄の記憶に新しいところだと思うが，それがC言語でもできてしまうのである。一般の高級言語のように使うこともでき，さらに，マシン語やBASICでなければ従来できなかった処理もこなせるようになっているのは，C言語の大きな特長である。

**●敵を知ってみると**

敵を知り，己を知れば……非常に敗色濃厚な気がする。Cコンパイラはこのような特長を備えたC言語をマシン語に変換する。したがって，プログラムの実行速度はマシン語と同等と考えていい。ただ，変換はけっこう汎用性を考えて行われるようになっているので，マシン語で直接記述したプログラムほど速度を上げることはできないのが救いといえば救いである。

また，C言語のほうがプログラムが読みやすいなどという意見もある。読者諸兄も雑誌などでこのような意見をご覧になったことがあろうかと思うが，この際はっきりしておこう。これは嘘である。どのような言語で書こうと，読みにくいプログラムは読みにくく，きたないプログラムはきたない。そんな理由でC言語を支持するのは馬鹿げている。

吾輩が敗色濃厚であると認めるのは，C言語の手軽さとマシン語の重さ，そしてマシン語に迫るその柔軟性である。C言語はまだマシン語に追いついてはいない。マシン語でなければできない処理というのは依然として存在する。たとえば数m秒以内に応答しなければならないプログラムはC言語では書けない。Cコンパイラによってどのようなマシン語に変換されるか（一般には）わからないため，実行にどの程度の時間がかかるのかも未知数だからである。このようなプログラムは1つひとつの命令の実行時間がわかっているマシン語を使って直接（時間を計算しながら）作ることになる。このように限られた分野においては少なからぬメリットはあるものの，一般のプログラムをマシン語で書く積極的な理由は見つけられない。先ほど救いとして挙げたプログラムの実行速度の問題だが，これもGCCの登場によって危うくなってきている。中級レベルのプログラマが書いたマシン語プログラムより，高速な質のよいマシン語プログラムをGCCが出力するためである。

おそらく御仁がC言語を使い続けているのもこのためであろう。マシン語でなければ書けないプログラムを，御仁が吾輩のために作ってくれたことはない。まァ，根が浮気な御仁のことだから，ある日突然C言語に愛想をつかしてマシン語に戻ってこないとも限らない。せっかくなので，ここでC言語の特長をいくつか挙げてみたいと思う。きっと今後，連載の中でC言語を使うこともあるだろう。

## 簡素な文法

C言語の特長のひとつとして，文法が簡素であることが挙げられる。X-BASICは条件を判定する命令「if」や繰り返しを指定する命令「for」，画面に文字を表示する命令「print」など，いくつかの命令を持っている。X-BASICはまだ命令の数が少ないほうで，X1やX1turbo，MZ-2500などのBASICは非常に多くの命令を持っている。命令の数が多いということは，命令の使い方を示す「文法」も多いということである。C言語は，プログラムの骨格を規定する「if」や「for」などの命令だけを残し，ほかはすべて「関数」としてしまうことにより，非常に簡素な文法を持つことに成功している。

**●変数の型**

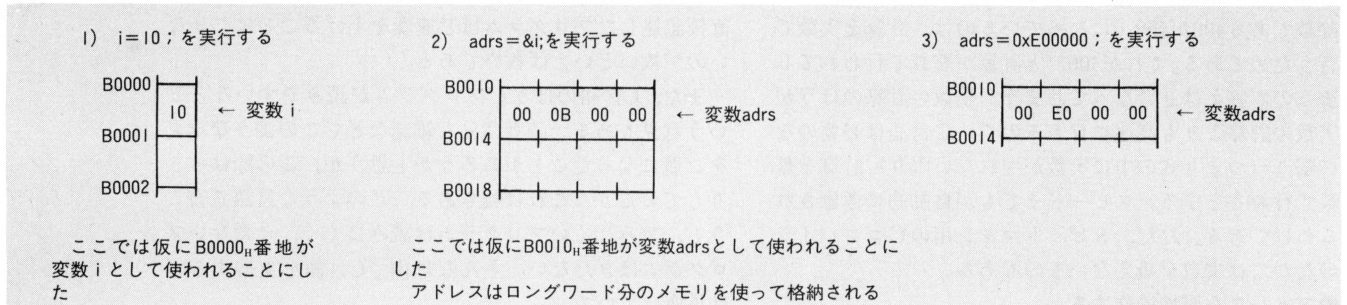
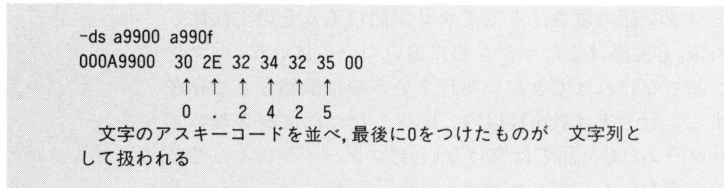
メモリからデータを取り出す際には，バイト単位に取り出すか，ワード（2バイト）単位にするか，はたまたロングワード（4バイト）単位とするかを指示した。C言

**図2　変数とメモリ**



**図3　文字列のメモリ内での表現**

```
-ds a9900 a990f
000A9900  30 2E 32 34 32 35 00
           ↑  ↑  ↑  ↑  ↑  ↑
           0  .  2  4  2  5
```

文字のアスキーコードを並べ,最後に0をつけたものが　文字列として扱われる

語もこれと同じように，使おうとする変数がデータを何バイト単位で扱うかを指示する方法を備えている。ただ，そこはそれ高級言語なので，バイトとかワードといった低レベルの用語は使わず，

1)　char　　バイト　　　　(character：文字)
2)　int　　　ロングワード　(integer：整数)
3)　short　　ワード　　　　(short int：短い整数)
4)　long　　ロングワード　(long int：長い整数)

といった単位を使う。読み方は順に「チャー」「イント」「ショート」「ロング」となる。3)，4)は，それぞれshort int，long intと表記しても構わない。

さらに実数用の型として，

5)　float　　4バイト　　　単精度実数
6)　double　8バイト　　　倍精度実数

の2つがある。これらは「フロート」「ダブル」と読む。

変数を使う際には，どのような名前の，どんなサイズのデータを扱う変数なのかプログラム中で宣言しなければならない。たとえば1バイトのデータを格納する i という名前の変数を宣言するなら，

char　i；

となる。この宣言をどこで行うかは後述する。

変数 i が仮に$B0000_H$番地に確保されたとすると，

i＝10；

と書けば，

move.b　#10,$B0000

が実行されることになり，図2-1のように$B0000_H$番地に10というデータがセットされる。

i*3.1416

なら$B0000_H$番地からデータが取り出されて，それが3.1416倍される。

実際に変数 i がどこに確保されたかを知りたければ，

&i

でアドレスを取り出すことができるのは前述のとおりである。C言語はこのアドレスを格納しておくための変数を宣言することもでき，これは，

char　＊adrs；

のように行う。これで，adrsという名前をつけた変数にアドレスを格納することができる。つまり，

adrs＝&i；

ということが可能なわけである。先の例に従えば，これでadrsには$B0000_H$がセットされることになる（図2-2)。変数 i のアドレスの代わりに，テキストVRAMの先頭アドレスである$E00000_H$を，

adrs＝0xE00000；

としてセットすることももちろんできる（図2-3)。

変数adrsにセットしたアドレスに入っているデータを扱いたい場合，すなわち$E00000_H$番地に入っているデータを扱いたい場合は，

＊adrs

のように変数の名前の前に‘＊’をつければいい。つまり，

printf( ”%d”, ＊adrs );

とすれば，$E00000_H$番地に入っているデータ(これは画面左上に表示されている8ドット分に相当する）が表示されることになる。先のprintfの例では“%g”を使っていたが今度は“%d”を使っている。“%g”は実数を表示する場合に，“%d”は整数を表示する場合に使用する。

逆に，

＊adrs＝0xFF;

とすれば，デバッガで$E00000_H$番地に$11111111_B$をセットした場合と同じように，画面左上に水色のラインが現れる。つまり先の，

char　＊adrs；

という宣言は，1バイトのデータを収めることのできる「＊adrs」という変数を宣言したものだと考えることもできる。この調子でいけば，int型のデータが入っているアドレスを格納する変数adrs1は，

int　＊adrs1；

で，long型のデータが入っているアドレスを格納する変数adrs2は，

long　＊adrs2；

で宣言できそうだが，まったくそのとおりである。

C言語ではアドレスを格納する変数のことをポインタ変数という。ポインタ（pointer）とは「指し示すもの」という意味である。このルールに従えば，変数adrsはchar型のデータが入っている場所を指し示すポインタ変数（ポインタを入れる変数）ということになる。C言語ではポインタは「アドレス」と同義である。なぜなら，C言語でアドレスという場合必ずそこにはなんらかのデータが入っており，「〜型のデータが入っている場所を指し示す」ものだからである。その意味で，

char　＊adrs；

のように変数adrsが宣言された場合,「adrsはchar型のデータへのポインタ」,あるいは「adrsはcharへのポインタ」であるといわれることが多い。

**●C言語プログラミングことはじめ**

C言語の最も簡単なプログラムは，次のような形をしている。

```
void main( void )
  {
     使用する変数の宣言
     計算式など
  }
```

よくC言語の入門書に次のようなプログラム,

```
void main( void )
  {
   printf( "Hello¥n" ) ;
  }
```

が掲載されている。これは変数宣言のない，C言語の最も単純なプログラムのひとつである。printf(……)の後ろに';'がついているのは,「文はセミコロン（；）で終わる」というC言語のルールによっている。

このプログラムを実行するには，まずCコンパイラを使えるようにしなければならない。XC Ver.2を例に説明しよう。XC Ver.2のシステムディスク1をセットして電源を投入する。すると，自動的にCコンパイラを使うのに必要なフロッピーディスク2枚が作成される。そのうちの「起動用ディスク」をディスクドライブにセットして電源を投入すれば，ごちごちゃと画面に文字が表示されたあと,

B>

と表示されて入力待ちになる。画面には「ドライブ0にランタイムディスクを，ドライブ1に作業ディスクをセットするように」との指示が出るが，ドライブ0には起動ディスクをセットしたまま，ドライブ1にプログラム作成用のディスクをセットして以下の作業を始める。

最初にやるべきことは，プログラムの作成である。これには，吾輩が携えたエディタというプログラム作成用のツールを使う。

B>ed test.c

とタイプしてリターンキーを押せば,test.cというファイルを作成する準備をしてED.Xというエディタが起動する。画面は真っ黒になり，画面下に水色でtest.cと表示されているはずである。これは現在test.cというファイルをエディット（編集）していますという合図だ。画面が真っ黒になったのは，test.cというファイルに(作り始めたばかりなので）なにも書いてないからである。

あとは雑誌のリストなどを見ながら，そのとおりにプログラムを入力していけばいい。次の行にいくにはリターンキー（↵ キー）を押す。行頭にスペースが入っている行はTABキーを押してスペースを入れる。間違えてしまったなら，カーソルキーを押してカーソルを移動し，BSキーかDELキーで削除すればいい。こうしてリストが入力できたら,「ESC」「E」の順にキーを押してED.Xを終了していただきたい。

B>dir

で,「test　c　……」というファイルができていることが確認できるだろう。ディスクの中身を見たければ，このように「dir」と入力すればいい。

プログラムが作成できたら，これをCコンパイラでマシン語に変換する。ドライブ0のディスクを「ランタイムディスク」に入れ替え,

B>cc test.c

とすればいい。間違いがなければ，しばらくしたあと再び,

B>

の状態に戻るはずである。

B>dir

と入力してみていただきたい。「test　x　……」というファイルができているはずである。これがマシン語に変換されたプログラムである。

実行するには,

B>test

と入力する。上の簡単なプログラムなら，画面に,

Hello

と表示されるであろう。

不幸にして間違いがあった場合，Cコンパイラは,

test.c　4　：Error　47：statement error.

などのメッセージを表示する。これは「test.cというファイルの4行目にstatement errorというエラーがある」という意味である。上のプログラムの3行目,

printf( "Hello¥n" ) ;

の最後の';'を忘れると，このメッセージにお目に掛かることができる。こんなときには,

B>cc test.c　>　err

と入力し，もう一度コンパイルしてみていただきたい。「>　err」というのは,「画面の代わりにerrというファイルに文字を書きなさい」という指示である。

コンパイルが終わったら,

B>ed err

で再びエディタを起動されたい。ED.Xはerrというファイルを読み込んで起動する。今度は画面は真っ黒ではなく，先ほど画面で見たのと同じメッセージが表示されているはずである。ここでおもむろに「ESC」「V」の順にキーを押すと，test.cが自動的に読み込まれ，4行目にカーソルが表示される。ここにエラーがあるというわけだ。いくつもエラーがある場合には，これは非常に楽な方法である。errファイルの見たいエラーの場所にカーソルを移動させ,「ESC」「V」の順にキーを押すだけで飛んでいってくれる。また,「ESC」「A」の順にキーを押すと，再び先のerrファイルに戻ることができる。

なぜ';'のない3行目ではなく4行目がエラーとなったのかは，少々説明しなければなるまい。Cコンパイラは，test.cというファイルを読み込みマシン語へ変換し始める。「printf(……)」をチェックしたところ';'がない。なんだまだ続くのかと次の行を見にいったらいきなり'}'で終わっているではないか。これは変だというので「statement error（記述が変だ)」となったわけである。プログラムを修正したら,「ESC」「E」でエディタを終了し，再びコンパイルしていただきたい。

## 画面に自分で文字を書く

前々回，御仁がデバッガを使ってテキストVRAMに直接データをセットしていたときのことをお話しした。ひとしきり遊んだあと，さすがにデバッガでデータをセットしていくのは面倒だと思ったのか，御仁はC言語でデータをセットするプログラムを作ったのである。今月はこのプログラムを紹介して幕としよう。

C言語ではポインタを使ってメモリを自由に扱えることを説明した。テキストVRAMとて例外ではない。少々C言語に通じている御仁は，「MC68000ではポインタは内部的にintと同じである」ということを逆手に取って，

＊0xE00000＝0x00；
＊0xE00080＝0x10；
……

のように整数0xE00000に‘＊’をつけていけば，データをセットできるのではないかと考えたのである。ものぐさな御仁ならではの発想といえよう。結果はエラーの山を築いただけであった。いかにC言語が柔軟とはいえ，整数とポインタくらいは（たとえ内部表現が同じであっても）区別するのである。

そして御仁が作ったのがリスト1のプログラムである。書き直しはさぞや面倒だっただろうと思うが，自業自得というもの。いい勉強である。さすがに今度はポインタを宣言し，ポインタを介してデータを書き込んでいる。

さて，コンパイル結果やいかに。実はこれはWarning（警告）の嵐となる。なぜならば，ポインタに整数をセットしているからである。さすがにメモリを直接扱うものだけにCコンパイラのほうも慎重で，「本当にそんなことしてもいいの？」とお伺いを立てているわけだ。御仁の様子はと見てみれば，「う～ん，律儀に警告を発してくれるねェ。でも，いいったらいいんだよ～ん」と素知らぬ顔でプログラムを実行する。

バスエラーが発生しました

……。御仁は大切なことを忘れていた。前回もお話ししたとおり，テキストVRAMはスーパーバイザ領域にあるので，ユーザーモードで動作するプログラムでは扱えないのである。

さて，三度目の正直がリスト2である。読者諸兄はこちらで試していただきたい。なお，便宜上行番号を付けてあるが，これは入力しなくていいので注意してほしい。ユーザーモードで動くプログラムをスーパーバイザモードで動くように変えるのが6行目である。このようにパラメータ0を指定すると，スーパーバイザモードになる。この行と23行はスーパーバイザに切り替える操作と再びユーザーに戻る操作として覚えておかれるといいだろう。

さて，7～22行には，(char ＊) という文字が付け加わっている。これはキャストと呼ばれる操作で，なんと，型を変えてしまう効果を持っている。0xE00000は整数だが，(char ＊) を付けることによって，charへのポインタへと型を変換しているのである。整数はこのようにポインタへと型を変換することができる。また，charへのポインタをintへのポインタへと型変換することも可能である。これらはCコンパイラが発する警告をなくすために入れられている。charへのポインタを，charへのポインタ変数に代入するのだから，警告の起きようはずもあるまい。

このプログラムを作ってから御仁は気づいた。

＊0xE00000＝0x00；

がエラーとなったのは，整数に‘＊’を付けたからであった。ならば，

＊(char ＊)0xE00000＝0x00；

でもよかったのではないか。まさにそのとおりである。

0xE00000 → 整数E00000H
(char ＊)0xE00000 → char型データが入っているアドレスE00000H
＊(char ＊)0xE00000 → その中身

というわけで，これでも正しくテキストVRAMにデータを書き込めるのである。御仁はこのプログラムも作ったのだが，こちらの実験は諸兄に任せるとしよう。なお，前々回のお楽しみデータをこのプログラムでやってみようという方は，charへのポインタをshortへのポインタ，あるいはlongへのポインタとしなければならない点に注意して試してみていただきたい。

次回は今回触れることのできなかった「関数」「簡潔なC言語の命令」編をお届けする予定である。とおり一辺倒の説明しかするつもりはないので，より詳細に知りたい方は中森氏の連載のほうで勉強されるとよかろう。

**リスト1　画面に自力で文字を表示する**

```
 1: void main( void )
 2: {
 3:         char    *adrs;
 4:
 5:         adrs = 0xE00000; *adrs = 0x00;
 6:         adrs = 0xE00080; *adrs = 0x10;
 7:         adrs = 0xE00100; *adrs = 0x28;
 8:         adrs = 0xE00180; *adrs = 0x44;
 9:         adrs = 0xE00200; *adrs = 0x82;
10:         adrs = 0xE00280; *adrs = 0x82;
11:         adrs = 0xE00300; *adrs = 0x82;
12:         adrs = 0xE00380; *adrs = 0x82;
13:         adrs = 0xE00400; *adrs = 0xFE;
14:         adrs = 0xE00480; *adrs = 0x82;
15:         adrs = 0xE00500; *adrs = 0x82;
16:         adrs = 0xE00580; *adrs = 0x82;
17:         adrs = 0xE00600; *adrs = 0x82;
18:         adrs = 0xE00680; *adrs = 0x82;
19:         adrs = 0xE00700; *adrs = 0x00;
20:         adrs = 0xE00780; *adrs = 0x00;
21: }
```

**リスト2　画面に文字を表示する（三度目の正直版）**

```
 1: void main( void )
 2: {
 3:         char    *adrs;
 4:         int     sp;
 5:
 6:         sp = SUPER( 0 );
 7:         adrs = (char *)0xE00000; *adrs = 0x00;
 8:         adrs = (char *)0xE00080; *adrs = 0x10;
 9:         adrs = (char *)0xE00100; *adrs = 0x28;
10:         adrs = (char *)0xE00180; *adrs = 0x44;
11:         adrs = (char *)0xE00200; *adrs = 0x82;
12:         adrs = (char *)0xE00280; *adrs = 0x82;
13:         adrs = (char *)0xE00300; *adrs = 0x82;
14:         adrs = (char *)0xE00380; *adrs = 0x82;
15:         adrs = (char *)0xE00400; *adrs = 0xFE;
16:         adrs = (char *)0xE00480; *adrs = 0x82;
17:         adrs = (char *)0xE00500; *adrs = 0x82;
18:         adrs = (char *)0xE00580; *adrs = 0x82;
19:         adrs = (char *)0xE00600; *adrs = 0x82;
20:         adrs = (char *)0xE00680; *adrs = 0x82;
21:         adrs = (char *)0xE00700; *adrs = 0x00;
22:         adrs = (char *)0xE00780; *adrs = 0x00;
23:         SUPER( sp );
24: }
```

Assembler

# 多角形の塗りつぶし

Murata Toshiyuki 村田 敏幸

ラインルーチンに続いては，これもまたグラフィックの必需品となる多角形の塗りつぶしです。手法として今回取り上げるのはソリッドスキャンコンバージョンと呼ばれるアルゴリズムで，基本を押さえればさまざまに利用できるでしょう。

今回は多角形の塗りつぶしを取り上げる。塗りつぶしというよりは，“中身の詰まった（solidな）多角形を描く”といったほうが正確かもしれない。輪郭を描いてから内側の1点を起点に色を塗っていくシードフィル（いわゆるペイント）ではなく，ソリッドスキャンコンバージョンという手法を使う。

ソリッドスキャンコンバージョンは，図形を水平な線分に細分し，その水平線分を並べていくことで描画する方法だ。Z'sSTAFFの閉曲線ペイントや，最近X68000で復活したグラフィックパッケージMAGIC（1991年5月号付録ディスク収録）に応用例を見ることができる。卑近な例では，ボックスフィルだって，たいていは水平線分を並べることで描画しているから，ソリッドスキャンコンバージョンといえなくもない。もちろん今回作るサブルーチンは，四角形に限らない任意の多角形を描ける。

参考文献としては，いつもの『実践コンピュータグラフィックス（日刊工業新聞社）』があり，また，本誌1989年7月号の特集に丹氏の的を射た解説が載っている。

## アルゴリズム

ソリッドスキャンコンバージョンのアルゴリズムには驚くような仕掛けはなにもない。図形の輪郭とスキャンライン（＝走査線，つまり画面の横1ライン）との交点を算出しては求めた点の間を水平線分でつないでいく，という処理を各スキャンラインに対して機械的に繰り返すだけだ（図1）。交点がいくつも存在するときには，x座標の小さい順に2点ずつ選んで，その2点間を結んでいくものとする。結果，ちょうど，スキャンラインを左から見ていって，交点が現れるたびに色を置くか置かないかを反転する感じになる。このため，図形の内側の閉じた領域は塗りつぶされない（図2）。

ソリッドスキャンコンバージョン自体は円の塗りつぶし描画などにも適用することができるが，今回作るのはあくまで多角形を描くルーチンであり，輪郭は直線だけからなることを前提にする。この場合，図形とスキャンラインとの交点は，すべての辺とスキャンラインとの交点を求めることで得られる。ここで，辺を処理する順序によっては交点が必ずしもx座標の小さい順に求まるわけではないから，求めた交点は適当なメモリ領域（スキャンラインバッファと呼ばれる）にいったんためておいて，全部揃ってからソートする必要がある。

アルゴリズムの大筋は以上のとおりのシンプルなものだが，実現にあたっては2，3気をつけなければならないことがある。

まず，各スキャンラインごとの交点の数は通常，偶数でなければうまくない。輪郭が閉じていない図形では，交点数が奇数のスキャンラインで図3のような不自然な結果を生じる。一番右の交点と対になる点がないため，色を置く/置かないのサイクルが尻切れで終わってしまうわけだ。この事態を避けるため，輪郭が閉じていない場合は描画サブルーチン側で最初の点と最後の点をつなぐ辺を補うことになるだろう。

輪郭が閉じていれば丸く収まるかというとそうではない。スキャンラインが多角形の頂点の上を通るときに交点の数が奇数になるケースがある。図4を見てもらいたい。各辺を独立したものとして扱い，個別にスキャンラインとの交点を求めると，頂点において多角形はスキャンラインと2度交わる。図4

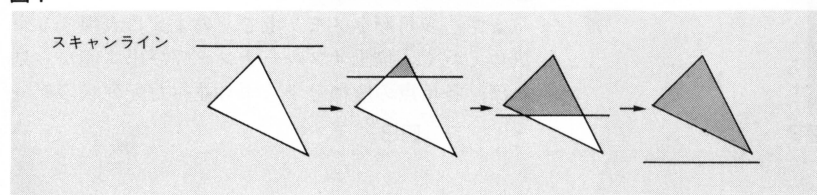
図1



図2

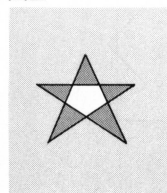



図3

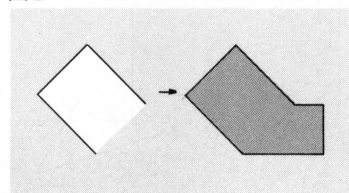



図4

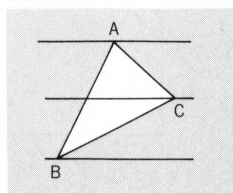



図5

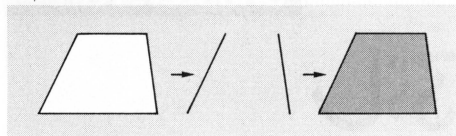



図6

| | | |
|---|---|---|
| +00 | IW | 頂点の個数(=n) |
| +02 | IW | 頂点1のx座標 |
| +04 | IW | 頂点1のy座標 |
| +06 | IW | 頂点2のx座標 |
| +08 | IW | 頂点2のy座標 |
| ⋮ | | |
| +?? | IW | 頂点nのx座標 |
| +?? | IW | 頂点nのy座標 |

の頂点Aの場合だと，辺ABとの判定で1回，辺CAとの判定で1回，計2回の判定により，点Aの座標がダブってスキャンラインバッファに格納される。頂点A，Bを通るスキャンラインでは，このダブりのお陰で交点の数が偶数に保たれ，期待どおりの描画結果が得られる。ところが，頂点Cを同じように交点2個にカウントしてしまうと正しい結果にならない。上向きや下向きではない頂点は交点1個と数えなければならないのだ。

残念ながら，この問題を回避するきれいな方法はない。真面目にやろうと思ったら，頂点が上下を向いているのか左右を向いているのかを前後の頂点との関係から判定して，泥臭くつじつまを合わせることになるだろう。多少不正確になってもかまわなければ，比較的ダメージの少ない手抜き手段がある。辺の長さを1ピクセル分短く見積もって，片方の端点についてはスキャンラインとの交わり判定を省略してしまうという方法だ。仮にY座標の大きなほうの端点を削る(その寸前で辺が終わるものと考える)ことにすると，上向きの頂点には辺が2本，下向きの頂点には0本，それ以外の頂点には1本の辺が集まることになる。線分の長さを誤魔化す分，下向きの頂点が欠けてしまうのが（文字どおり）欠点だが，とりあえず，交点の数のバランスだけは保たれる。

頂点以外では，水平な辺の扱いにも注意がいる。水平な辺はスキャンラインと重なってしまい，ほかの辺と同じように交点を求めるわけにはいかない。どう対処するのが正しいかというと，無視するのが正解だ。水平な辺は前処理段階で削ってしまう。削除しても，その辺を挟む辺同士が水平線分で結ばれるから，描画結果にはまったく影響しない（図5）。

では，プログラムの作成に向けてさらに細かな点を煮詰めていこう。

## 多角形の内部表現

まず，多角形をメモリ上でどのように表現するか決めておく。描画サブルーチンを呼び出す側から見れば，各頂点の座標をずらずら並べた配列状のデータとして多角形を用意し，サブルーチンに引き渡すという形が楽そうに思う。隣り合った2点が辺を形作り，また，最後の点は最初の点と結ばれているものと考える。実際にはデータの長さがわからないと困るので，先頭に頂点の個数を格納するとしよう。今回は，頂点の個数格納用に2バイト，1個の頂点の座標格納用に4バイト（x，y座標に各2バイト）の図6の形式に決める。座標の範囲が−32768〜＋32767の65535角形までを表現できるというわけだ。

さて，いま決めた頂点の列形式の多角形表現は，ソリッドスキャンコンバージョンの処理過程ではあまり扱いやすいとはいえない。効率よくスキャンラインと辺との交点が求められるよう，処理中は，頂点ではなく辺を単位に扱いたい。そこで，描画に先立って，頂点の列から辺のリストを作成することを考える。辺のリストの構造については少しあとで述べよう。

## クリッピング

クリッピングすることも当然考慮しておく。各辺のクリッピングには線分描画のときと同じ手法が使えるが，多角形の場合は，クリッピングしても頂点同士の関係（どの頂点とどの頂点が辺で結ばれているか）が変わらないよう注意する必要がある。多角形が頂点の列の形で表現されている場合は，並べられたままの順序で頂点をクリッピングしていけば，少なくとも頂点を結ぶ順序が乱れることはない。図7に示した三角形を頂点A，B，Cの順にクリッピングすると，点A，a，b，Cが得られ，ごく自然に点bとcをつなぐ辺が補われる形となる。

ところが，やはりといおうか，穴はある。図8左側の三角形をクリッピングすると点A，a，bしか得られない。aとbを結んでしまうと，結果は妙なことになる。クリッピングウィンドウの隅の点を補えればよいのだが，元の図形の辺上にない点を補うのには工夫が要る。また，クリッピングすることによって2つ以上の部分に分割される図形では，注意しないと分割した図形をつなぐ辺が勝手に生成されることもある。図8右側の凹多角形の場合，クリッピングすると点D，a，b，F，c，dが求まるが，これらの点をぐるりとつなぐと，本当はあってはならない辺abが生まれてしまう。

図8左側の三角形については，段階を踏んでクリッピングすることで正しい結果を求めることができる。まず，クリッピングウィンドウの1辺（の延長線）でクリッピングし，中間形を作ってから，順次残りの辺でクリッピングしていけばよい。また，一般解ではないが，ソリッドスキャンコンバージョンのためのクリッピングであれば，複数に分割される図形の場合も2段階のクリッピングで対処できる。前処理段階ではY座標についてのみクリッピングを行い，描画の時点になってからX座標でクリッピングする。この2回目のクリッピングは，スキャンラ

図7

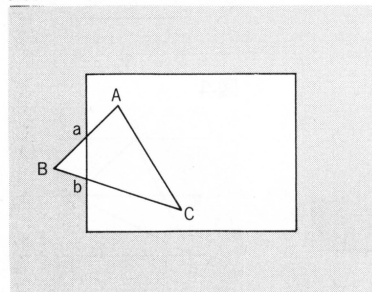
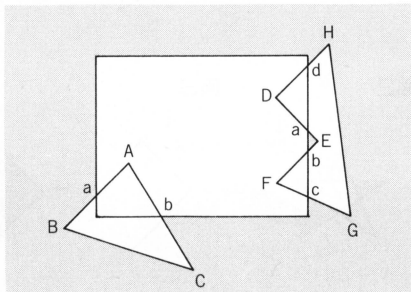



図8

インと図形との交点を求めてその交点間を結ぶとき，描画のぎりぎり直前に行う。これにより，“幻の垂直辺”は生まれなくなる。幻の水平線はまだ残るが，クリッピング後，水平線は削除するから，やはり問題にはならない。

## 多角形とスキャンラインとの交点の求め方

辺とスキャンラインの交点は数学的な手法で計算してもよいが，もっと効率のよい方法がある。Bresenhamの線分描画アルゴリズムを使って，端から順に求めていくのだ。線分描画のときは求めた座標に直接点を打ったのに対し，今度の場合，得られた座標はスキャンラインバッファに格納し，すべての辺との交点が得られた時点でソートして描画することになる。

ここで，スキャンラインの本数分のスキャンラインバッファが用意できれば，あらかじめ多角形とスキャンラインとの全交点を求めて記憶しておくことも可能だが，あまりにメモリの無駄遣いなので，スキャンラインバッファはあくまで1本分に抑えて使い回したい。となると，同時には1本のスキャンラインとの交点しか格納しておく余地がないので，Bresenhamのアルゴリズムに必要なパラメータ（誤差項やその増分など）は各辺ごとに構造体様のデータにまとめておき，複数の辺に対して並行してBresenhamのアルゴリズムを適用する形になる。

今回はリスト1のEDGE.Hで定義するような構造体を使った。いい加減につけたラベル名から，以下この構造体をEDGBUF構造体と呼ぶことにする。EDGBUFの構造はあとのプログラムに都合のよいよう，項目の位置や順序を決めてあるので，将来の改良時に備えて注釈をつけてある。

ここで特に重要なのはx座標を格納する項目を先頭にもってきている点だ。EDGBUF構造体のアドレスをアドレスレジスタに入れてアクセスする場合，通常，各項目はディスプレースメントつきアドレスレジスタ間接形式でアクセスすることになるが，先頭の1項目だけは，単純なアドレスレジスタ間接形式で少しだけ高速にアクセスできる。そこで，もっともよく参照されるx座標の項目を先頭にしてあるというわけだ。

さて，スキャンラインを上から順に処理する都合上，交点計算には無条件にyについてループする形でBresenhamのアルゴリズムを適用する。本来Bresenhamのアルゴリズムでは線分の傾きに応じてxでループするかyでループするか切り替えなければならないのだから，つじつま合わせが必要だ。図9のようななだらかな線分を上から処理していく場合，1ステップBresenhamのアルゴリズムを適用しても，x座標が減るだけでy座標は変化しない。そこで，y座標が変化するまで，強引にx座標を進めてしまう。結果，交点としては，図中黒で示した飛びとびの点が選択される。得られる点は片側に偏ってしまうが，今回は目をつぶった。

もう少し具体的な処理に踏み込もう。

すべての辺が常にスキャンラインと交わるわけではないから，交点の算出は“スキャンラインと交わることがわかっている辺”に対してだけ行う必要がある。この問題はEDGBUFをアクティブな（活性化した）ものと未アクティブなものに分けて処理することで解決する。前提として，EDGBUFを作成する際に，始点がかならず終点より上にくるようにしておく。初期状態では，各EDGBUFは未アクティブな群にまとめられる。そして，スキャンラインと始点のy座標が一致した時点でアクティブな群に移される。また，アクティブだったEDGBUFはスキャンラインが終点のy座標に一致した時点で削除される。交点の算出は，このアクティブなEDGBUF群に対してのみ行う。

処理上は，EDGBUFを群にまとめたりしなくてもフラグを設ければ十分だ。このフラグは初期化時に未アクティブにセットされ，やがてアクティブになり，最後には処理ずみの印へと変わる。が，今回はあえてフラグ方式はとらず，実際に群にまとめる方法をとった。未アクティブなEDGBUF，アクティブなEDGBUFそれぞれにつき，おのおののEDGBUFを指すポインタの配列を用意する。未アクティブな群からアクティブな群への移動は，このポインタの配列の上で行う。図10にイメージを示しておこう。

## 交点のソート

ソートアルゴリズムはべつになにを使ってもいい。多角形の辺の数が何10本，何100本とあるのでなければ，単純なアルゴリズムで十分だ。今回は単純挿入法を採用した。ただし，少しだけ細工がある。

これまでの話だと，スキャンラインと辺の交点はスキャンラインバッファに格納して，ソートはその中で行うことになる。が，あとで示すプログラムにはスキャンラインバッファが表立っては出てこない。求めた交点のx座標はそのままEDGBUF構造体に格納しておいて，EDGBUF構造体ごとソートする。正確には，ソートは構造体そのものではなく構造体へのポインタ配列に対して適用する。データを交換しなければならないときには，データ自体は動かさずにポインタだけをつなぎ替えるわけだ。アクティブかどうかをフラグで表さずにポインタ配列を用意したのはこのためだ。

わざわざこんな複雑（でもないが）なやり方を採

図9

図10

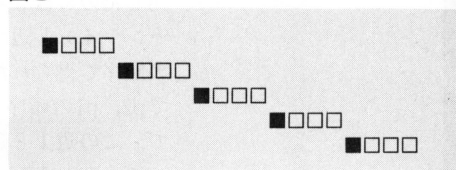
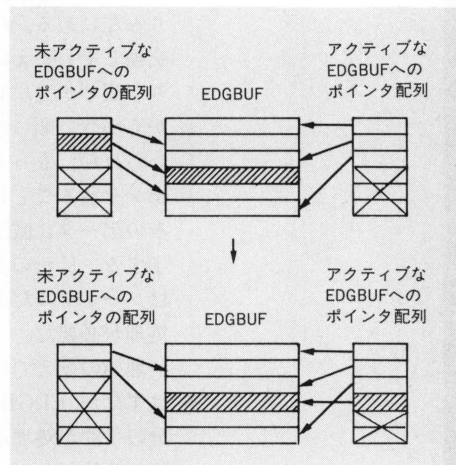
未アクティブなEDGBUFへのポインタの配列

EDGBUF

アクティブなEDGBUFへのポインタ配列

未アクティブなEDGBUFへのポインタの配列

EDGBUF

アクティブなEDGBUFへのポインタ配列

用したのにはもちろん理由がある。いま，あるスキャンラインと辺1，辺2が交わり，交点p1，p2がこの順序で得られたとする。p2<p1の場合は並べ替えてp2，p1の順序にしてから描画するわけだ。ここで，この辺1と辺2の位置関係は次のスキャンラインでも変わらない確率が高いということに注目しよう。たぶん，次のスキャンラインでも辺2は辺1よりも左にある。ソートのときにEDGBUF構造体ごと交換してしまえば，次のスキャンラインでは辺1よりも辺2が先に処理されることになり，交点は最初から小さい順に求まる。すでにソートする必要がないわけだ。もっとも処理の都合上，毎回ソートルーチンを通ることにはなるが，単純挿入法はソートずみのデータに滅法強く，あっという間にソートを完了する。ほかの多くのソーティングアルゴリズムではソートずみだということを認識するまでに余計な処理が必要だ。

理屈のうえでは，この方法はなかなか効率がよいはずだが，EDGBUFをポインタで間接的に参照する分の手間が処理速度を落とす可能性もある。で，実際に2種類のプログラムを作って速度を比べてみた。ランダムな多角形を描かせた場合，四～五角形ぐらいまでは，アクティブかどうかをフラグで表しソートはスキャンラインバッファ上で行う単純な方法のほうがほんの少し速かった。描く図形の大きさにもよるが，五～六角形ぐらいでとんとんになり，以降はEDGBUFのポインタ配列を用意してソートもポインタ配列上で行うほうが速くなる。今回選択した方法は，速くするためというよりは，辺の数が増えてもなるべく遅くしないための工夫といえるだろう。

## プログラムの実際

というあたりでプログラムを見てもらおう。考えがあって，プログラムは大きく3つのサブルーチンに分割した。第1のサブルーチンgcilppolyでは描く多角形の（Y座標に関する）クリッピングのみを行う。第2のサブルーチンgenedgelistはクリッピング後の頂点列から先ほど示した構造の辺リストを作成する。第3のサブルーチンgfillpolyはこの辺リストを参照して実際に多角形を描画する。これら3つのサブルーチンを続けて呼び出すことでひとつの多角形が描かれる。

このような構成にしたのは，処理単位を明確にするという教育的意味も少しあるが，変則的な図形に対応できるようにするためでもある。メインルーチン側で工夫すれば，輪郭が一筆書きできない（たとえば穴の空いた）図形も描けるようになっている。図形の輪郭を構成する個々の多角形を別データとして用意し，個別に辺リストの作成までを行ってから，作成した辺リストをまとめてgfillpolyに渡せばよい。また，同一の図形を色だけ変えて再描画する場合（たとえば，描いた図形を消す場合），gclippoly，genedgelistの呼び出しは1回だけですむというメリットもある。

**●クリッピング部（リスト2）**

多角形のクリッピングルーチンgclippolyには，引数として頂点の列データと，クリッピング後の頂点座標を格納するメモリ領域のアドレスを渡す。クリッピングによって角が削ぎ落とされると最大で頂点の数が2倍に増えることになるから，結果を受け取るメモリはそれを見越して大きめに確保しておかなければならない。

クリッピング時には，処理中の頂点と直前に処理した頂点の関係によって適当に処理を振り分けている。クリッピングはまずウィンドウの上端，つづいて下端の2回に分けて行っている。ある点pをウィンドウの1端でクリッピングする場合，直前に処理した点（p0とおく）とpとの関係には次の4通りが考えられる。

1)　pもp0も不可視側にある
2)　pは不可視側にあり，p0は可視側にある
3)　pは可視側にあり，p0は不可視側にある
4)　pもp0も可視側にある

明らかに，両方が不可視側にある1)のケースではpは単に捨てられる。2)のケースではpをウィンドウ端にクリップし，得られた点をバッファに登録する。3)のケースではまずp0をウィンドウ端にクリップして得られた点を登録してからpも登録する。両方とも可視側にある4)のケースではpのみを登録する。p0は直前のループで登録ずみのはずだからだ。

なお，リスト2はちょっと手抜きで，直前の点が可視だったかどうかを覚えておかずに，毎回，可視だったか不可視だったかを調べ直している。多少，効率を落としているかもしれない。手抜きといえば，例の“辺の終点を無視する”都合で，ウィンドウ下端でのクリッピングを1ピクセル甘くしてある。きっちりウィンドウの下端でクリッピングしてしまうと，終点を無視する副作用で，ウィンドウの最下ラインにはまったく描画が行われなくなってしまうのだ。

また，処理単位を明確にするとかいっておきながら，リスト2ではどさくさに最初の頂点と最後の頂点を結ぶ処理を行っている（118～120行）。

**●辺リストの作成部（リスト3）**

辺のリストを作成するgenedgelistにはクリッピング後の頂点列と，辺リスト格納用のメモリ領域アドレスを渡す。ここでは，隣り合った頂点を結ぶ辺をBresenhamのアルゴリズムで描くための誤差項そのほかのパラメータの初期値を計算し，EDGBUF構造体に格納する。このとき，水平な辺は見つけしだい削除している（26～27行）。辺リストの作成がすんだら，genedgelistはa0レジスタにその最終アドレスを入れて戻る。作成した辺リストの個数を返してもよかったのだが，穴の空いた図形を描く場合など，複数の輪郭を一連の辺リストにまとめたいというときには，最終アドレスがわかったほうが便利だと判断した結果だ。

●**描画部（リスト4）**

実際に描画を行うgfillpolyには，引数として，

辺リストの先頭アドレス
辺リストの最終アドレス
作業用（EDGBUFのポインタ配列格納用）のメモリアドレス
描画色のパレットコード

を並べたメモリ領域のアドレスを渡す。辺リストの最終アドレスにはgenedgelistからの戻り値をセットすることになる。

gfillpoly自体は処理ごとにさらにいくつかのサブルーチンに分割してある。最初に一度だけ呼び出される初期化ルーチンinit（194〜238行）では，辺の数を数えながら，すべての辺を未アクティブな群にまとめ（204〜213行），アクティブな辺の群を空にし（219行），レジスタの初期化も行う。204〜213行のループではついでにy座標の最小値を求めたりもしている。このy座標で表されるスキャンラインから処理を開始することにより，どの辺とも交わらないスキャンラインの処理を省くことができる。

ここでは，辺の群(EDGBUFのポインタの配列)の管理の方法に注目してもらいたい。ポインタ配列の先頭アドレスはa3レジスタで示される。a3の指すアドレス以降に未アクティブなEDGBUFへのポインタが並び，末尾はa4レジスタで示される。そして，a4レジスタの指すアドレス以降につづけてアクティブなEDGBUFへのポインタを並べ，末尾をa5で示す。ある辺がアクティブになったら，適当にポインタをつなぎ替え，境界であるa4をずらしていく。ポインタ配列1本分のスペースで2本分の働きを実現し，使用メモリを節約しているのだ。

あと，初期化ルーチンのもうひとつのポイントはアクティブなEDGBUFのポインタ配列の末尾に番人を2個立てている点だ。この番人はソート時と描画時の両方に使われる。2個目の番人は，描画時に交点が奇数個になった場合のストッパーとして働く。ちなみに番人は頭のx座標を格納する項目しかないEDGBUF構造体として，240行で用意している。

初期化がすんだら50行からのメインループに制御が移る。ループ内では，まず未アクティブなEDGBUF群から始点が現在処理中のスキャンライン上にあるものを探して，あればアクティブにする（51〜53行）。それから，アクティブなEDGBUFを交点のx座標でソートし，描画して，次のスキャンラインとの交点を求める(55〜60行)。交点を求めるときには，同時に処理ずみのEDGBUFの削除も行っている。スキャンラインが画面下に達するまで，この処理を繰り返せば多角形が描かれる。もっとも，リスト4では，スキャンラインのy座標ではなく，EDGBUFの残り個数を終了条件に使っている。EDGBUFがなくなったら，それ以上の処理は必要ないわけだ。

アクティブにすべきEDGBUFを探すサブルーチンactivate（75〜90行）ではEDGBUFをアクティブ化するときのポインタのつなぎ替えと境界の移動（85〜86行）の仕方が小さなポイントだが，ほかには特に見るべき点はない。ソートするサブルーチンsortlist (95〜112行）も，扱うデータが生のデータではなくポインタになっている点を除けば，以前に作ったサブルーチンと同じ形をしているのがわかると思う。なお，sortlistでは，番人があらかじめセットされていることを前提にしている。

1ライン描画するサブルーチンdrawline（116〜151行)では，得られた交点のx座標を画面の左右端でクリッピングしながら水平線分の描画を行っている。始点がウィンドウの右端より右にあったら以降の点も不可視だから処理を打ち切る（125〜126行)。また，終点がウィンドウの左端より左にあったら始点も不可視だから，その線分は描かずスキップする(130〜131行)。どちらでもなければ，線分は少なくとも部分的に可視だから，gmacro.hで定義したマクロMINとMAXを使って両端点がウィンドウ内に収まるのを保証してから実際に描画する。例によって水平線分の描画部分はループを展開してある。

次のスキャンラインとの交点を求めるサブルーチンcalcnextxでは，アクティブな各EDGBUFごとにBresenhamのアルゴリズムの1ステップを施し，x座標と誤差項eを更新している。傾きがなだらかな辺も強引にyでループしている関係で，yが変化するまで(誤差項eが負になるまで)，169〜171行のループでx座標を進めている。終点に達したかどうかは，y座標ではなく，線分の長さ−1を初期値とするカウンタ（EDGBUFの項目DY）で判定している。

## 動作試験

最後に動作試験用のプログラムをリスト5に示しておく。リスト5はキーが押されるまで，ランダムな七角形を描画し続ける。座標の範囲を128〜383に制限して図形のサイズを限定した“はったりデモモード”になっているので，ペイントしか知らない人が見たら驚くぐらいの速度にはなっていると思う。頂点の数は6行のラベルNMAXPOINTで指定しているから適当に変更してみるといい。また，71〜72行を殺して代わりに73行を復活すれば座標の範囲が0〜511になる。40〜42行を復活してクリッピングウィンドウを設定し，クリッピングがうまく働いているかどうかも確認しておいてほしい。

*

今回のプログラムでは，描画アルゴリズムそのものよりも使用したデータ構造が少々難解だったかもしれない。ひとことアドバイスするなら，このテの込み入ったデータ構造は，具体的なメモリイメージと抽象的なイメージとを頭の中で切り替えながら見ていくと理解しやすいはずだ。

次回はまだ流動的だが，そろそろ，拡大・縮小とか回転とかをやらなければならないかな，と思っている（弱い予告)。

## リスト1　EDGE.H

```
 1:           .offset 0
 2: *
 3: *         EDGBUF構造体
 4: *
 5: *MEMO:
 6: *genedgelist, gfillpolyはXが先頭にあることを仮定している
 7: *gfillpolyは
 8: *         XとDEY   DYとE    DY0とE0
 9: *が並んでいることを仮定している
10: *
11: X:        .ds.w   1          *スキャンラインとの交点のx座標
12:                              *(初期値はX0)
13: DEY:      .ds.w   1          *誤差項の補正値 dy*2
14: DEX:      .ds.w   1          *誤差項の増分     dx*2
15: DY:       .ds.w   1          *ループカウンタ
16:                              *(初期値はDY0)
17: E:        .ds.w   1          *誤差項           e
18:                              *(初期値はE0)
19: SX:       .ds.w   1          *傾きの符号       sgn(x1-x0)
20: X0:       .ds.w   1          *始点のx座標      x0
21: Y0:       .ds.w   1          *始点のy座標      y0
22: DY0:      .ds.w   1          *線分の長さ       y1-y0
23: E0:       .ds.w   1          *誤差項の初期値 e0
24: EDGBUFSIZ:
25:
26:           .text
```

## リスト2　GCLIPPOLY.S

```
  1: *          多角形をcliprectで指定された矩形領域で
  2: *          クリッピングする
  3:
  4:            .xdef   gclippoly
  5:            .xref   cliprect
  6: *
  7:            .offset 0
  8: *
  9: MINX:      .ds.w   1
 10: MINY:      .ds.w   1
 11: MAXX:      .ds.w   1
 12: MAXY:      .ds.w   1
 13: *
 14:            .text
 15:            .even
 16: *
 17: gclippoly:
 18: POINTS  =          8
 19: POINTS2 =          12
 20:            link    a6,#0
 21:            movem.l d0-d7/a0-a5,-(sp)
 22:
 23:            movem.l POINTS(a6),a1-a2
 24:                                    *a1 = クリッピング前の点
 25:                                    *a2 = クリッピング後の点
 26:            lea.l   cliprect,a0     *a0 = クリッピングウィンドウ
 27:
 28:            addq.l  #2,a2           *点の数を格納する2バイトを
 29:                                    *  飛ばす
 30:
 31:            move.w  (a1)+,d7        *d7 = 点の数
 32:            subq.w  #1,d7           *
 33:            bcc     do              *
 34:
 35:                          *1点しか与えられていない場合
 36:            movem.w (a1),d0-d1
 37:            cmp.w   MINY(a0),d1     *y < MINY なら
 38:            blt     done            *  不可視
 39:            cmp.w   MAXY(a0),d1     *y > MAXY なら
 40:            bgt     done            *  不可視
 41:            move.w  d0,(a2)+        *(x,y)を登録する
 42:            move.w  d1,(a2)+        *
 43:            bra     done            *  クリッピング完了
 44:
 45:                          *2点以上与えられていた場合
 46: do:        moveq.l #0,d0
 47:            move.w  d7,d0
 48:            add.l   d0,d0
 49:            add.l   d0,d0
 50:            movem.w 0(a1,d0.l),d2-d3
 51:                                    *(d2,d3) = 最後の点
 52:                                    *        = 最初の点にとっては
 53:                                    *          直前の点
 54:            movea.w #$8000,a5       *a5 = ゲタ
 55:            bra     lpent
 56:                          *クリッピングする点を(x,y)
 57:                          *直前の点を(x0,y0)とおく
 58: loop:      movem.w (a1)+,d2-d3     *(d2,d3) = (x0,y0)
 59: lpent:     movem.w (a1),d0-d1      *(d0,d1) = (x,y)
 60:            moveq.l #0,d4           *d4 = 直前の点が可視かどうかのフラグ
 61:                                    *     (仮に可視と考える)
 62:
 63: cminy:     move.w  MINY(a0),d5     *d5 = MINY
 64:            cmp.w   d5,d1           *y < MINYなら
 65:            blt     cminy0          *  (x,y)をMINYでクリップ
 66:
 67:            cmp.w   d5,d3           *y >= MINY かつ y0 >= MINY なら
 68:            bge     cmaxy           *  MINYでのクリッピングは不要
 69:
 70:                          *(x,y)は可視で(x0,y0)は不可視
 71:            exg.l   d0,d2           *(x0,y0)をMINYでクリップ
 72:            exg.l   d1,d3           *
 73:            bsr     minyclip        *
 74:            exg.l   d0,d2           *
 75:            exg.l   d1,d3           *
 76:            moveq.l #-1,d4          *(x0,y0)をクリッピングした
 77:            bra     cmaxy
 78:
 79: cminy0:    cmp.w   d5,d3           *y < MINY かつ y0 < MINY なら
 80:            blt     next            *  辺は完全不可視
 81:
 82:                          *(x,y)は不可視で(x0,y0)は可視
 83:            bsr     minyclip        *(x,y)をMINYでクリップ
 84:
 85: cmaxy:     move.w  MAXY(a0),d5     *d5 = MAXY
 86:            addq.w  #1,d5           *d5 = MAXY+1 = MAXY'
 87:                                    *(gfillpolyの手抜き対応)
 88:            cmp.w   d5,d1           *y > MAXY'なら
 89:            bgt     cmaxy0          *  (x,y)をMAXY'でクリップ
 90:
 91:            cmp.w   d5,d3           *y <= MAXY' かつ y0 <= MAXY'なら
 92:            ble     setp            *  MAXY'でクリッピングは不要
 93:
 94:                          *(x,y)は可視で(x0,y0)は不可視
 95:            exg.l   d0,d2           *(x0,y0)をMAXY'でクリップ
 96:            exg.l   d1,d3           *
 97:            bsr     maxyclip        *
 98:            exg.l   d0,d2           *
 99:            exg.l   d1,d3           *
100:            moveq.l #-1,d4          *(x0,y0)をクリッピングした
101:            bra     setp0
102:
103: cmaxy0:    cmp.w   d5,d3           *y > MAXY' かつ y0 > MAXY'なら
104:            bgt     next            *  辺は完全不可視
105:
106:                          *(x,y)は不可視で(x0,y0)は可視
107:            bsr     maxyclip        *(x,y)をMINYでクリップ
108:
109: setp:      tst.w   d4              *(x0,y0)をクリッピングしたのなら
110:            beq     setp1           *
111: setp0:     move.w  d2,(a2)+        *  クリッピングした(x0,y0)も
112:            move.w  d3,(a2)+        *  登録する
113: setp1:     move.w  d0,(a2)+        *クリッピングした(x,y)を登録する
114:            move.w  d1,(a2)+        *
115:
116: next:      dbra    d7,loop         *すべての点について繰り返す
117:
118:            movea.l POINTS2(a6),a1  *最初の点を
119:            move.w  2(a1),(a2)+     *  バッファ最後に書き込み
120:            move.w  4(a1),(a2)+     *  図形を閉じる
121:
122: done:      movea.l POINTS2(a6),a1  *クリッピング後の
123:            addq.l  #2,a1           *
124:            suba.l  a1,a2           *
125:            move.l  a2,d0           *
126:            lsr.l   #2,d0           *  点の数を数える
127:
128:            swap.w  d0              *点の数が65536以上なら
129:            tst.w   d0              *
130:            beq     setn            *
131:            moveq.l #0,d0           *  誤動作防止に0にしてしまう
132:
133: setn:      swap.w  d0              *
134:            move.w  d0,-(a1)        *点の数を記録する
135:
136:            movem.l (sp)+,d0-d7/a0-a5
137:            unlk    a6
138:            rts
139: *
140: maxyclip:                    *MAXY'でクリッピングする
141:            cmp.w   d5,d3           *反対側の端点がMAXY'上なら
142:            beq     just            *  その点が求める点
143:
144:            move.w  d2,a3           *反対側の端点の
145:            move.w  d3,a4           *  座標を保存
146:            add.w   a5,d0           *両端点とMAXY'に
147:            add.w   a5,d1           *  8000hのゲタを履かせて
148:            add.w   a5,d2           *  無符号化する
149:            add.w   a5,d3           *
150:            add.w   a5,d5           *
151:
152:                          *クリッピングする点を(x1,y1)
153:                          *反対側の点を(x2,y2)
154:                          *両者の中点を(mx,my)とおく
155: maxylp:    move.w  d1,d6           *
156:            add.w   d3,d6           *
157:            roxr.w  #1,d6           *d6 = my = 中点のy座標
158:            cmp.w   d5,d6           *my = MAXY'なら
159:            beq     yclipq          *  クリッピング完了
160:            bcs     maxy0
161:                          *y1 < my < MAXY' < y2
162:            move.w  d6,d1           *y1 = my
163:            add.w   d2,d0           *
164:            roxr.w  #1,d0           *x1 = mx
165:            bra     maxylp          *  と更新して繰り返す
166:
167:                          *y1 < MAXY' < my < y2
168: maxy0:     move.w  d6,d3           *y2 = my
169:            add.w   d0,d2           *
170:            roxr.w  #1,d2           *x1 = mx
171:            bra     maxylp          *  と更新して繰り返す
172: *
173: minyclip:                    *MINYでクリッピングする
174:            cmp.w   d5,d3
175:            beq     just
176:
177:            move.w  d2,a3
178:            move.w  d3,a4
179:            add.w   a5,d0
180:            add.w   a5,d1
181:            add.w   a5,d2
182:            add.w   a5,d3
183:            add.w   a5,d5
184:
185: minylp:    move.w  d1,d6
186:            add.w   d3,d6
187:            roxr.w  #1,d6
188:            cmp.w   d5,d6
189:            beq     yclipq
190:            bcc     miny0
191:
192:            move.w  d6,d1
193:            add.w   d2,d0
194:            roxr.w  #1,d0
195:            bra     minylp
196:
```

▶X68000を買って1年。「初心者のための環境構成術」を読んで「よかった，もう俺は初心者じゃないんだ」とつくづく思ってしまいました。　田中　正憲(16)神奈川県

```
197: miny0:  move.w  d6,d3
198:         add.w   d0,d2
199:         roxr.w  #1,d2
200:         bra     minylp
201:
202: yclipq: move.w  d6,d1               *d1 = 求めたy座標
203:         add.w   d2,d0               *
204:         roxr.w  #1,d0               *d0 = それに対応したx座標
205:
206:         sub.w   a5,d0               *ゲタを脱がせる
207:         sub.w   a5,d1               *
208:         move.w  a3,d2               *反対側の点の
209:         move.w  a4,d3               *  座標を復帰
210:         rts
211: *
212: just:   move.w  d2,d0               *反対側の点が
213:         move.w  d3,d1               *  ちょうど求める点
214:         rts
215:
216:         .end
```

## リスト3 GENEDGELIST.S

```
 1: *       辺のリストを作成する
 2: *       (ソリッドスキャンコンバージョンの前処理)
 3:
 4:         .include        edge.h
 5: *
 6:         .xdef   genedgelist
 7: *
 8:         .text
 9:         .even
10: *
11: genedgelist:
12: POINTS  =       8
13: EDGLST  =       12
14:         link    a6,#0
15:         movem.l d0-d4/a1-a2,-(sp)
16:
17:         movem.l POINTS(a6),a0-a1
18:         exg.l   a0,a1
19:
20:         move.w  (a1)+,d4            *d4 = 点の数
21:         subq.w  #2,d4               *
22:         bcs     gdone               *最低2点必要
23:
24: geloop: movem.w (a1),d0-d3          *2点の座標を取り出す
25:
26:         cmp.w   d1,d3
27:         beq     genext              *水平の辺は無視する
28:
29:         bgt     gnedg1              *y0 < y1を保証する
30:         exg.l   d0,d2               *
31:         exg.l   d1,d3               *
32: gnedg1: move.w  d0,X0(a0)           *X0(a0) = x0
33:         move.w  d1,Y0(a0)           *Y0(a0) = y0
34:
35:         sub.w   d3,d1               *d1 = y0-y1 = -dy
36:         move.w  d1,E0(a0)           *E0 = -dy
37:
38:         neg.w   d1
39:         move.w  d1,DY0(a0)          *DY0(a0)= y1-y0 = dy
40:
41:         add.w   d1,d1
42:         move.w  d1,DEY(a0)          *DEY(a0)= dy*2
43:
44:         moveq.l #0,d1               *sgn(x1-x0)と
45:         sub.w   d0,d2               *abs(x1-x0)を求める
46:         beq     gnedg3              *
47:         bpl     gnedg2              *
48:         moveq.l #-1,d1              *
49:         neg.w   d2                  *
50:         bra     gnedg3              *
51: gnedg2: moveq.l #1,d1               *
52:
53: gnedg3: move.w  d1,SX(a0)           *SX(a0) = sgn(x1-x0)
54:
55:         add.w   d2,d2
56:         move.w  d2,DEX(a0)          *DEX(a0)= dx*2
57:
58:         lea.l   EDGBUFSIZ(a0),a0
59:
60: genext: addq.l  #4,a1               *1点分進める
61:         dbra    d4,geloop
62:
63: gdone:  movem.l (sp)+,d0-d4/a1-a2
64:         unlk    a6
65:         rts
66:
67:         .end
```

## リスト4 GFILLPOLY.S

```
  1: *       ソリッドスキャンコンバージョン
  2:
  3:         .include        gconst.h
  4:         .include        gmacro.h
  5:         .include        edge.h
  6: *
  7:         .xdef   gfillpoly
  8:         .xref   gramadr
  9:         .xref   cliprect
 10: *
 11:         .offset 0
 12: *
 13: MINX:   .ds.w   1
 14: MINY:   .ds.w   1
 15: MAXX:   .ds.w   1
 16: MAXY:   .ds.w   1
 17: *
 18:         .offset 0
 19: *
 20: EDGES:  .ds.l   1           *辺リスト先頭アドレス
 21: EDGEED: .ds.l   1           *辺リスト最終アドレス
 22: EDGARY: .ds.l   1           *EDGBUFのポインタ配列用ワーク
 23: COL:    .ds.l   1           *描画色
 24: *
 25:         .text
 26:         .even
 27: *
 28: gfillpoly:
 29: ARGPTR  =       4
 30:         movem.l d0-d7/a0-a6,-(sp)
 31: SAVREGS =       8+7
 32: ARGPTR  =       ARGPTR+SAVREGS*4
 33:         movea.l ARGPTR(sp),a0   *a0 = 引数受け渡し領域
 34:         bsr     init            *初期化する
 35:         bra     lpent
 36:
 37: *       d0.l    描画色パレットコード(上位/下位ワードとも)
 38: *       d1~d4   作業用
 39: *       d5.l    G-RAM1ラインのバイト数(=GNBYTE)
 40: *       d6.w    残りEDGBUF数
 41: *       d7.w    注目しているスキャンラインのy座標
 42: *
 43: *       a0~a2   作業用
 44: *       a3      未処理EDGBUFのポインタ配列先頭
 45: *       a4      未処理EDGBUFのポインタ配列末尾
 46: *               (=処理中EDGBUFのポインタ配列先頭)
 47: *       a5      処理中EDGBUFのポインタ配列末尾
 48: *       a6      注目しているスキャンラインの左端アドレス
 49:
 50: mainloop:
 51:         cmpa.l  a4,a3               *未処理のEDGBUFが
 52:         beq     nomore              *  もうなければスキップ
 53:         bsr     activate            *アクティブにすべきEDGBUFを探す
 54:
 55: nomore: cmpa.l  a5,a4               *処理中のEDGBUFが
 56:         beq     nextln              *  なければスキップ
 57:
 58:         bsr     sortlist            *交点のx座標でソートする
 59:         bsr     drawline            *1ライン描画する
 60:         bsr     calcnextx           *つぎの交点を求める
 61:
 62: nextln: adda.l  d5,a6               *a6 = つぎのラインの左端アドレス
 63:         addq.w  #1,d7               *d7 = つぎのラインのy座標
 64:
 65: lpent:  tst.w   d6                  *EDGBUFが残っているあいだ
 66:         bne     mainloop            *  繰り返す
 67:
 68:         movem.l (sp)+,d0-d7/a0-a6
 69:         rts
 70:
 71: *       未処理EDGBUF群の中から
 72: *       注目しているスキャンライン上に始点があるものを探し,
 73: *       あれば, 処理中EDGBUFのポインタ配列に移動する
 74: *
 75: activate:
 76:         movea.l a3,a1               *a1 = 未処理EDGBUFのポインタ配列
 77: actvlp: movea.l (a1)+,a0            *a0 = EDGBUFへのポインタ
 78:         cmp.w   Y0(a0),d7           *始点のy座標 = 現在のy座標?
 79:         bne     actvnx              *  でなければ, まだ
 80:
 81:                             *EDGBUFをアクティブにする
 82:         move.w  X0(a0),(a0)         *最初の交点のx座標(= x0)を設定
 83:         move.l  DY0(a0),DY(a0)      *ループカウンタと誤差項を初期化
 84:
 85:         move.l  -(a4),-(a1)         *未処理EDGBUFのポインタ配列から削除
 86:         move.l  a0,(a4)             *処理中EDGBUFのポインタ配列に追加
 87:
 88: actvnx: cmpa.l  a4,a1               *すべての未処理EDGBUFについて
 89:         bcs     actvlp              *  繰り返す
 90:         rts
 91:
 92: *       交点のx座標の小さい順に
 93: *       処理中のEDGBUFのポインタ配列をソートする
 94: *
 95: sortlist:
 96:         move.l  a5,d2               *a5を待避
 97:
 98:         subq.l  #4,a5               *単純挿入法でソートする
 99:         bra     sortnx              *
100: srtlp2: move.l  a1,-8(a2)           *
101: srtlp1: movea.l (a2)+,a1            *
102:         cmp.w   (a1),d1             *
103:         bgt     srtlp2              *
104:         move.l  a0,-8(a2)           *
105: sortnx: movea.l a5,a2               *
106:         movea.l -(a5),a0            *
107:         move.w  (a0),d1             *
108:         cmpa.l  a4,a5               *
109:         bcc     srtlp1              *
110:
111:         movea.l d2,a5               *a5を復帰
112:         rts
113:
114: *       1ライン分描画する
115: *
116: drawline:
117:         movea.l a4,a1               *a1 = 処理中のEDGBUFのポインタ配列先頭
118:         lea.l   hline0(pc),a2       *
119:
120:         move.w  cliprect+MINX,d3        *d3 = MINX
121:         move.w  cliprect+MAXX,d4        *d4 = MAXX
122:
123: drawlp: movea.l (a1)+,a0            *a0 = EDGBUFへのポインタ
124:         move.w  (a0),d1             *d1 = 描く水平線分の始点x座標
```

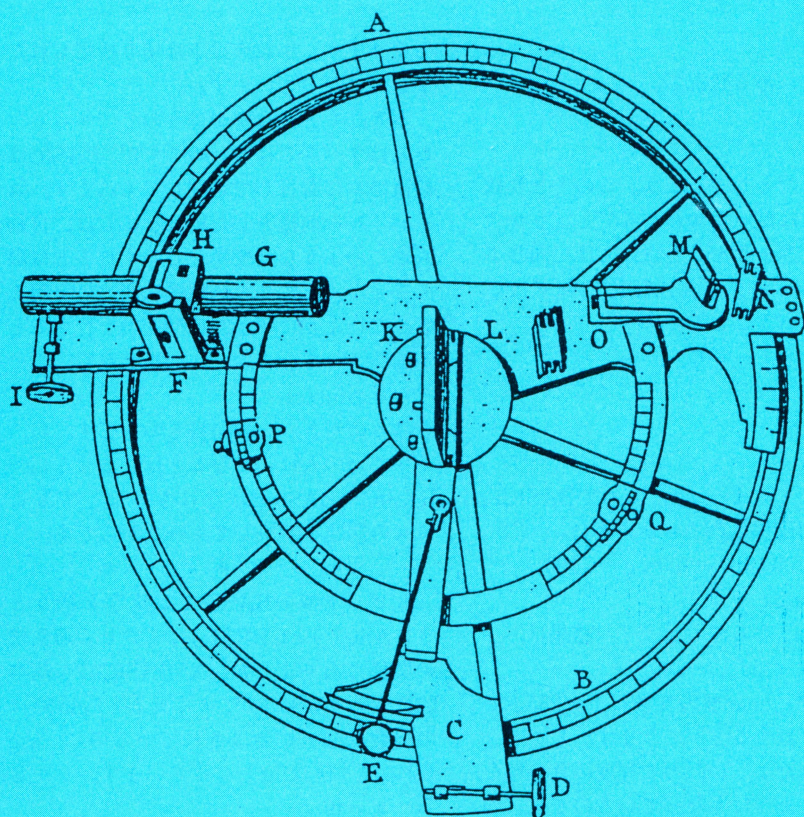
▶「PC-9801のマウスを使う」はとてもいいですね。私はすでに2つも死んだマウスを持っているので非常に助かります。　藤原　常雅(20)神奈川県

```
125:         cmp.w   d4,d1               *始点 > MAXXならば
126:         bgt     drawq               *  線分は画面の右外だから描画完了
127:
128:         movea.l (a1)+,a0            *
129:         move.w  (a0),d2             *d2 = 描く水平線分の終点x座標
130:         cmp.w   d3,d2               *終点 < MINXならば
131:         blt     drawlp              *  線分は画面の左外
132:
133:         MAX     d3,d1               *始点を画面左端でクリップ
134:         MIN     d4,d2               *終点を画面右端でクリップ
135:         sub.w   d1,d2               *d2 = 描く水平線分の長さ-1
136:
137:         add.w   d1,d1               *
138:         lea.l   0(a6,d1.w),a0       *a0 = 描く線分の左端アドレス
139:         addq.w  #1,d2               *d2 = 描く線分の長さ
140:         bclr.l  #0,d2               *奇数?
141:         beq     pxeven              *
142:         move.w  d0,(a0)+            *  奇数ピクセルの分
143: pxeven: neg.w   d2                  *
144:         jmp     0(a2,d2.w)          *水平線分描画
145:
146: hline:  .dcb.w  GNPIXEL/2,$20c0 *move.l d0,(a0)+
147:
148: hline0: cmpa.l  a5,a1
149:         bcs     drawlp
150:
151: drawq:  rts
152:
153: *       つぎのスキャンラインおける
154: *       交点のx座標を求める
155: *
156: calcnextx:
157:         movea.l a4,a1               *a1 = 処理中EDGBUFのポインタ配列先頭
158: calclp: movea.l (a1)+,a0            *a0 = EDGBUFへのポインタ
159:         subq.w  #1,DY(a0)           *カウンタを減らす
160:         beq     deactivate          *  0になったら配列から削除する
161:                                     *  (最後の1点は処理しない)
162:         move.w  E(a0),d1            *d1 = e
163:         add.w   DEX(a0),d1          *e += 2*dx
164:         bmi     calskp              *e < 0ならx座標は今回のまま
165:
166:         move.w  SX(a0),d4           *sgn(x1-x0) = 0なら
167:         beq     calcnx              *  x座標は不変
168:         movem.w (a0),d2-d3          *d2 = x, d3 = 2*dy
169: inclp:  add.w   d4,d2               *x += sgn(x1-x0)
170:         sub.w   d3,d1               *e -= 2*dy
171:         bpl     inclp               *e >= 0のあいだ繰り返す
172:
173:         move.w  d2,(a0)             *求めたx座標
174: calskp: move.w  d1,E(a0)            *eを更新
175:
176: calcnx: cmpa.l  a5,a1               *すべての処理中EDGBUFについて
177:         bcs     calclp              *  繰り返す
178:         rts
179: *
180: deactivate:
181:         move.l  -(a5),-(a1)         *用ずみのEDGBUFを削除する
182:         move.l  4(a5),(a5)          *その分番人を詰める
183:         subq.w  #1,d6               *残りEDGBUF数を減らす
184:         bra     calcnx
185:
186: *       初期化
187: *
188: *       ・すべてのEDGBUFを未処理EDGBUFのポインタ配列にまとめる
189: *       ・処理中EDGBUFのポインタ配列を空にする
190: *       ・最初に注目するスキャンラインのy座標と左端アドレスを求める
191: *       ・EDGBUFの数を数える
192: *       ・描画色をレジスタに設定する
193: *
194: init:
195:         movem.l (a0)+,a1-a3         *a1 = EDGBUF配列の先頭
196:                                     *a2 = EDGBUF配列の末尾
197:         movea.l a3,a4               *a3 = a4
198:                                     *   = 未処理EDGBUFのポインタ配列先頭
199:         moveq.l #-1,d1              *d1 = 仮の最小y座標
200:         moveq.l #0,d6               *d6 = EDGBUFのカウンタ
201:         move.w  (a0)+,d7            *d7 = 描画色
202:         bra     ilpent
203:
204: initlp: move.l  a1,(a4)+            *順に未処理EDGBUFのポインタ配列に追加
205:         move.w  Y0(a1),d0           *始点のy座標が
206:         cmp.w   d1,d0               *  仮の最小y座標より小さければ
207:         bcc     notmin              *
208:         move.w  d0,d1               *  仮の最小y座標を更新する
209:
210: notmin: addq.w  #1,d6               *EDGBUFの数を数え上げる
211:         lea.l   EDGBUFSIZ(a1),a1         *a1 = つぎのEDGBUF
212: ilpent: cmpa.l  a2,a1               *すべてのEDGBUFについて
213:         bcs     initlp              *  繰り返す
214:
215:                                     *d1 = 最小のy座標
216:                                     *d6 = EDGBUFの個数
217:                                     *a4 = 未処理EDGBUFのポインタ配列末尾
218:                                     *   = 処理中EDGBUFのポインタ配列先頭
219:         movea.l a4,a5               *a5 = 処理中EDGBUFのポインタ配列末尾
220:                                     *(a4 = a5だから空)
221:
222:         lea.l   dmydat(pc),a0       *番人を置く
223:         move.l  a0,(a5)             *(交点が奇数個の場合に備え
224:         move.l  a0,4(a5)            *  番人は2個)
225:
226:         moveq.l #0,d0               *x = 0
227:         bsr     gramadr             *
228:         movea.l a0,a6               *a6 = 最初に処理する
229:                                     *       スキャンラインの左端アドレス
230:         move.l  #GNBYTE,d5          *d5 = G-RAM1ラインのバイト数
231:
232:         move.w  d7,d0               *d0 = 描画色
233:         swap.w  d0                  *
234:         move.w  d7,d0               *
235:
236:         move.w  d1,d7               *d7 = 最初に処理する
237:                                     *       スキャンラインのy座標
238:         rts
239: *
240: dmydat: .dc.w   $7fff               *番人として使うダミーデータ
241:                                     *(XのフィールドしかないEDGBUF)
242:
243:         .end
```

## リスト5 POLYTEST.S

```
  1: *       gfillpolyのテスト用プログラム
  2: *
  3:         .include        doscall.mac
  4:         .include        edge.h
  5: *
  6: NMAXPOINT       equ     7           *頂点の最大数(2~32767)
  7: NMAXEDGE        equ     NMAXPOINT*2+1   *辺の最大数
  8: *
  9:         .xref   gfillpoly
 10:         .xref   gclippoly
 11:         .xref   genedgelist
 12:         .xref   setcliprect
 13: *
 14: FPACK   macro   callno
 15:         .dc.w   callno
 16:         endm
 17: *
 18: __RAND          equu    $fe0e
 19: *
 20:         .offset 0
 21: *
 22: EDGES:  .ds.l   1           *辺リスト先頭アドレス
 23: EDGEED: .ds.l   1           *辺リスト最終アドレス
 24: EDGARY: .ds.l   1           *EDGBUFのポインタ配列用ワーク
 25: COL:    .ds.l   1           *描画色
 26: *
 27:         .text
 28:         .even
 29: *
 30: ent:
 31:         lea.l   inisp,sp
 32:
 33:         move.l  #$0010_0005,-(sp)
 34:         DOS     _CONCTRL            *512x512, 65536色モード
 35:         addq.l  #4,sp
 36:
 37:         clr.l   -(sp)               *スーパーバイザ
 38:         DOS     _SUPER              *  モードへ
 39:
 40: *       pea.l   window(pc)          *クリッピング
 41: *       jsr     setcliprect         *  ウィンドウを
 42: *       addq.l  #4,sp               *  設定する
 43:
 44:         lea.l   argbuf,a1
 45:         pea.l   (a1)
 46: loop:   bsr     setarg
 47:
 48:         pea.l   edges
 49:         pea.l   pnts2
 50:         pea.l   pnts
 51:         jsr     gclippoly           *クリッピングして
 52:         addq.l  #4,sp
 53:         jsr     genedgelist         *辺リストを作成し
 54:         addq.l  #8,sp
 55:         move.l  a0,EDGEED(a1)       *辺リスト最終アドレス
 56:         jsr     gfillpoly           *描画する
 57:
 58:         DOS     _KEYSNS             *キーが押されるまで
 59:         tst.l   d0                  *
 60:         beq     loop                *  繰り返す
 61:
 62:         DOS     _INKEY              *キーを読み捨てる
 63:
 64:         DOS     _EXIT
 65: *
 66: setarg:
 67:         lea.l   pnts,a0
 68:         move.w  #NMAXPOINT,(a0)+
 69:         move.w  #NMAXPOINT*2-1,d1
 70: arglp:  FPACK   __RAND
 71:         lsr.w   #7,d0               *0≦d0≦255
 72:         addi.w  #128,d0             *128≦d0≦383
 73: *       lsr.w   #6,d0               *0≦d0≦511
 74:         move.w  d0,(a0)+
 75:         dbra    d1,arglp
 76:
 77:         FPACK   __RAND
 78:         move.w  d0,COL(a1)
 79:         rts
 80: *
 81:         .data
 82:         .even
 83: *
 84: argbuf: .dc.l   edges               *gfillpolyへの引数
 85:         .dc.l   edges               *
 86:         .dc.l   edgary              *
 87:         .dc.w   0                   *描画色
 88: *
 89: window: .dc.w   64,64,511-64,511-64
 90: *
 91:         .bss
 92:         .even
 93: *
 94: pnts:   .ds.w   1                   *クリッピング前の頂点
 95:         .ds.l   NMAXPOINT
 96: pnts2:  .ds.w   1                   *クリッピング後の頂点
 97:         .ds.l   NMAXEDGE            *
 98: edges:  .ds.b   NMAXEDGE*EDGBUFSIZ          *EDGBUFの配列
 99: edgary: .ds.l   NMAXEDGE+2          *EDGBUFのポインタ配列
100:                                     *(番人2個の分を含む)
101: *
102:         .stack
103:         .even
104: *
105:         .ds.l   2048
106: inisp:
107:         .end    ent
```

［特集］

# Personal tool, BASIC

たとえば，なにかのツールを使っているとき，こんな機能があったら……という思いは誰しも抱くものです。パソコンでの処理であれば，アセンブラが使えればアセンブラで，Ｃ言語が使えればＣ言語で必ず解決できます。どうしてBASICではできないと思われているのでしょうか？
GCCの登場はX68000の開発環境に革命をもたらしました。開発の中心がアセンブラからＣ言語へ移向しつつあるのも当然でしょう。しかし，同じ恩恵がX-BASICにももたらされているということは忘れられがちになっています。
X-BASICに対する不満は少なくありませんが，考えてみればＣ言語と同程度には構造化でき，インタプリタとコンパイラの環境を持ち，DOSと親和性の高いコマンドを作成でき，そしてGCCがある場合には世界最高水準の最適化が施されるのです。BASICを学ぶことは百害あって一利なしといった言葉はX-BASICにはまったくあてはまりません。
言語の複雑さは必要なマニュアルの量に比例します。誰にでも扱えるBASICはパーソナルユースでは最強のツールといえるでしょう。

CONTENTS

文法と基本作法

# X-BASICの基礎

Nakano Shuichi 中野 修一

**手軽さゆえにパソコンでは依然として幅広いユーザーを持つBASIC。しかしX-BASICは独自の環境を持っています。「BASICなら使えるでもX-BASICはよくわからない」という人のためのX-BASIC入門です。**

「ほかのBASICのつもりでプログラムを書いたらX-BASICでつまずいた」という方も多いと聞きます。

まず，BASICという名前にだまされてはいけません。パソコン上の多くのBASICが低級言語指向の性格を持っているのに対し，X-BASICは（C言語に近いといわれながらも）遙かに高級言語を指向しています。根本的に「違う言語」といってもいいでしょう。それを頭に入れておいてください。

では，本論に入りましょう。

## パソコンでプログラミングをする

いいつくされたことですが，プログラミング技術があっても目的を持たない人にはプログラムは作れません。実際になんらかのプログラムを作る際にはプログラミング言語の知識などより，それ以外にどんな知識を持っているかということのほうが重要になります。極論すれば（経験的な事実として），マニュアル1冊あればプログラムの知識は不要です。

はっきりした目的を持っている人にはここでは多くを語るつもりはありません。自らの道を進んでください。

それ以外に，なにかの処理をしていて突然発生する要求があります。そういったものに対処する専用のツールがあればそれにこしたことはないのですが，そういうときのためにプログラム言語は「ツール」として役立ちます。

簡単にコマンドの作成ができるC言語はそういった傾向が強いですし，AWKという言語などは最たるものといっていいでしょう。そしてC言語でできることの多くはX-BASICでだってできます。状況によってはインタプリタのほうが有利なことも往々にしてあるものです。とっさのときに役に立つ，なんでもできるツール，それがBASICの一面だともいえるでしょう。

では，X-BASICに特徴的な概念からまとめてみることにしましょう。

## 型

たとえば，ある状況で，

```
print a
100
```

と表示されたとしましょう。このとき変数aの型はなんだったかわかるでしょうか？ 実は判定できません。逆にいえば，同じように見えるデータでもBASIC内部で扱い方が違うということです。

ダイレクトモードで，

```
int a
char b
float c
```

と入力したとしましょう。これでBASIC内部に変数が確保されました。次に，

```
a=−100:b=−100:c=−100
print a,b,c
```

としたらどうなるでしょう？　結果は，

```
−100    156    −100
```

となります。char型の変数は0～255までのデータを表すことができます。しかし，ここに表れたように，ある程度の負の数を受け付けるようにもなっているのです。内部では−128～−1の値は256を足した128～255と同等のものとして扱われます。

char型はX-BASICでもっとも小規模なデータを扱う型で主に文字コードを格納するときに使用されます。半角文字の文字コードは0～255の範囲に収まりますから，ほかのデータ型を使う場合に比べてメモリ消費が少なくてすみます。

*　　　*　　　*

次にfloat型である変数cに円周率を示す関数pi( )の値を代入してみましょう。

```
c=pi( )
print c
3.1415926535898
```

となりますね。ここで，

```
a=c
print a
```

とすると結果は，

```
3
```

となります。小数点以下は切り捨てられたかたちになっていますね。

本来，aとcは型が違うので“a=c”などはすべきではないのですが，BASICは数値のみに着目して処理してくれます。コンパイルを前提とした場合などでは，自動的にこのような型変換をしてくれるとは限りません。ですから処理系に間違いを起こさせないように明示的に型変換を行うことが推奨されています。この場合なら，

```
a=int(c)
```

とするのがきっと教科書どおりです。が，たいていの人はこんなことはしません。X-BASICやBAStoCはなにもしなくてもこれらの問題を解決してくれるからです。

*　　　*　　　*

8ビット機のBASICでは文字を扱うときにASC( )とCHR$( )はまさしく逆変換の関係にありました。X-BASICでも基本的に変わりませんが，よく見ると，chr$( )の引数はcharが要求されているのに，asc( )の戻り値はintとなっています。しかし，

```
a=49
print chr$(a)
```

でもまったく支障ありません（aはint）。charは一定の範囲内のintとまったく同じように扱われます。ほかの型にはそれぞれの型変換関数が用意されていますが，intとcharのあいだにはなにもありません。

同様に1文字ずつデータを扱うfputc( )とfgetc( )を見てみましょう。これもfgetc( )の引数はchar，fputc( )の戻り値はintとなっています。fgetc( )は読み込んだコードを返す関数ですし，文字コードは0～255に決まっていますからchar型が適しているように思えますね。しかし，fgetc( )はエラー発生時には−1を返します。これはcharでは255と区別できません。ですから通常の文字処理ではデータの受け渡しに

intが使われます。エラーを－1で返すというのはかなりの関数で共通した作法となっているのです。

## X-BASIC特有のもの

続いてほかのBASICでは見られないステートメントを拾ってみましょう。

**●switch**

まずはswitch文。これはもともとC言語の文法の範疇にあるもので，ほかのBASICには見られない構造です。

```
switch a
    case 1
        [ A ]
      break
    case 2
        [ B ]
    default
        [ C ]
endswitch
```

という構造があったとしましょう。これはaが1のときAの処理，2のときはBの処理，それ以外のときはCの処理を……，というのは誤りで2のときはBに続いてCの処理も行います。

aによってそれぞれ処理を分けようとすると，いちいちbreakを書かねばなりません。世間一般のcase構文とは動作が違うのでとまどう人もいるかもしれませんね(SLANGやPASCALをやった人くらいかな？)。

switch文の実体は，aの値によって，それぞれ対応するcase部分にgotoしているだけなのです。よって途中で止めないとどんどん次の処理を行います。途中のcase～はラベルにすぎません。

これを逆手に取って，

```
switch a
    case 0
    case 1
    case 5
        [ A ]
      break
    case 2
    case 7
        [ B ]
        ⋮
endswitch
```

のように不規則な条件をまとめてしまうこともできます(もっとも，K&Rではひとつのcaseごとにbreakをつけるように指導されています)。

**●break**

さて，ここで出てきたbreak文もほかのBASICとは違うものです。

先の例を見てもわかるとおり，switchやループの途中から抜け出すためのステートメントです。これはforループの中のforループとかswitch途中のswitchとか，入り組んだ状況では1段階ずつ脱出します。

無制限なところに飛んではいかないシステム推奨のgoto文といえるでしょう。内部処理的にもgotoと違い，スタックの積み残しのない安全設計となっています。

**●continue**

似たようなものにcontinueがあります。これは英語の意味からでは動作が予測しにくい文といえるでしょう（break文の動作を復旧するものと思ったのは私だけでしょうか)。マニュアルには「for，while，repeat文の次のループを強制的に開始します」と

### プログラムの組み方

基本的にプログラムの組み方に決まったものはありません。それぞれの人が自分流にプログラムを組む，これがパーソナルコンピューティングの姿です（プロは別です)。以前，

「美しいプログラムは動くプログラムだ」

と書いたことがありました。動かないことには綺麗でも話にならないことと，ちゃんと動いているのに「スパゲティだから」とソースを公開しない人を念頭に置いた言葉でした。

「スパゲティもできたてはおいしい」

という名言も残されています。プログラムの質を云々する以前に1本のプログラムを完成しきる力量をつけることが第一ですから。プログラムがちゃんと動いているというのは，それなりに凄いことなのです。経験的に見て初心者ほど「大作」に挑みがちです。なんでもいいから「完成品を作る能力」をつける。これは結構困難なことです。そのためなら手段は問うべきではない，というのが私の意見です。

しかし，基本的な知識を持っておくのは無駄ではないと思います。以下にプログラミングの一般論（？）をまとめてみましょう。

**●構造化とは**

最近ではあまりこだわる人もいないのですが，X-BASICで「いかにもBASIC」といったプログラムを見るとなにか悲しいものがあります。

よく耳にする言葉，「構造化」とは，プログラムをブロック分けをしていくことが基本になります。もっとも重要なことは「各ブロックの入りロと出口をひとつずつにする」ということでしょう。goto文をなくすとか，字下げをするとかいったことは二の次です。

そして「各ブロックの持つ機能をできるだけ独立かつ単純化すること」が数多くの経験が教えるプログラミングの結論です。

**●入りロと出口の統一**

多くの場合，プログラムは関数の集合体として記述されます。関数を使った場合，呼び出した関数は呼び出されたところに戻ってきますから，各部をブラックボックス化しやすいといえるでしょう。

内容的には確かにそうです。関数を多用していれば，なにも考えなくてもある程度プログラムは構造化されてきます。しかし，どんなスパゲティプログラムでも処理は流れます。ある程度しか構造化されていないということが忘れられがちになっているようです。プログラムは結局「読む」ものですから，見た目が重要視されるのもいたしかたないでしょう。

ここで大原則を挙げておきます。なんらかの構造（たとえばループ）に途中から入ることはいかなる場合にも許されません。途中から出るのは原則的に許されます。つまり「入りロひとつに出口は複数」というのがまっとうなプログラムで容認される構造です。しかし，たいていの場合，出口もひとつにできるものです。「入りロと出口の統一」が構造化の第一段階です。

X-BASICではgotoやgosubが事実上使えないため，構造の途中に飛び込んでくるプログラムはほとんど書けなくなっています。

ここでもう一度X-BASICの関数を見てみましょう。入りロは統一されますが，出口のほうは文法どおりでもreturn( )とendfuncの2種類があります。関数を「関数」として使う場合はreturn( )，「手続き」として使う場合はendfuncが終端となります。PASCALなどでは区別されるプログラム構造を一緒に扱っているための構造的な弊害です。

ですから，X-BASICやC言語で出口を形式的に統一することにどれほどの意味があるのかはわかりません。実際，X-BASICの入門書でも構造化プログラムの例として関数のあちこちに出口を持ったプログラムが載っていたりします。しかし，構造化の題目のひとつ「順次処理，選択，繰り返し構造だけでプログラムが構成できる」という定理は「適正プログラム（入りロがひとつで出口がひとつのプログラム）は」という前提の下にのみ証明されたものなのです。

そのほかにも，比較的多用される複数の出口を持つパターンは次のようなものでしょう。

```
func test(a)
    switch a
        case 0: return(0)
        case 1: return(1)
    endswitch
    return(2)
endfunc
```

プログラムを読むときにswitch文のような構造を見たら処理が分散していることを念頭に置いておきましょう。

endはプログラム中でいくつ使ってもよいとマニュアルには書いてあります。しかし，1回の起動で一度しか実行されないとわかりきっているものを複数置くのは無駄な気がしますね。コンパイラにかけるとき誤動作の原因にもなります。「存在は必要以上に増やしてはならない」という格言もあります。プログラムにおいては処理の流れを単純にするように配慮することが大切です。

他人に見せることはなくても1カ月後の自分は他人と同じですから，他人にわかりやすいプログラムを書くというのが基本でしょう。

あります。いまひとつイメージははっきりしません。ここで,

```
repeat
    print a
    a=a+1
     if a=10 then continue
until a=10
```

というプログラムがあった場合, continueで次のループが強制的に開始されると無限ループになってしまいますね。でも実際にはこのプログラムはちゃんと止まります。

continueの動作は, そのループの繰り返し判定部にジャンプするだけなのです。その場所からループの終わりまでの処理を飛ばしたいときに使います。

## X-BASICでのツール作り

BASICはインタプリタとしての美点を持っています。しかし, ちゃんとX-BASICを使おうと思うとCコンパイラを持っていることが最低条件となります。コンパイルできるか否かでプログラム言語の評価はまったく変えなければなりませんから。

従来の「BASIC=インタプリタ→遅い」「C言語=コンパイラ→速い」といった単純な図式はひとまず忘れてください。コンパイルされたBASICのプログラムは必要な程度には速くなります。

さて, X-BASICの場合, どんなプログラムでも完全にコンバートできるとは限らないので, いったん完成したBASICプログラムをコンバートしようとすると結構てこずることがあります。しかし, コンパイルを前提に開発すればことは簡単です。テストコンパイルを何回か繰り返せば, 確実に動作するプログラムを仕上げていくことができます。これは雑作もないことです。XBAStoC CHECKERという製品も発売されています。これは徹底的にX-BASICの構文チェックをやってくれますのでコンパイラを使わない人にもおすすめできるツールです(どうしてもバグが取れないときには)。

## コンパイルに伴う処理

コンパイルしたプログラムはコマンドラインから使うことが多いと思われます。そういった場合,

```
b_init();
```

という行を削除するほうが使いやすいことも往々にしてあります。

これはおそらくデフォルトのキーや画面初期設定などを行っているものと思われるので, 必要な場合にはプログラム側で初期化などの対応をすることにしましょう。

さらに,

```
b_exit(0);
```

という行を,

```
_exit(0);
```

に変えると処理終了時の画面初期化などを実行しなくなります。

これでコンパイルしたプログラムはコマンドラインから実行しても違和感なく使えるようになります。

ついでに, コンパイルを前提としたときの特典として,「コマンドラインからのパラメータ渡し」という技があります。

bc.xにかけられたプログラムには必ず,

```
main(b_argc,*b_argv[])
```

という部分が見受けられます。これらは, BASIC上では,

b_argcはパラメータの個数+1

b_argv()は文字列型配列でそれぞれのパラメータを表します(ただしインタプリタ上で動かしても意味はない)。

```
A>test aa bb cc dd
```

というふうにコマンド入力された場合,

```
b_argc=5
b_argv(0)=test
b_argv(1)=aa
b_argv(2)=bb
b_argv(3)=cc
b_argv(4)=dd
```

のようになります。コマンド名自体もパラメータの数に入っていることを頭に入れておけば, 間違えることはないでしょう。

気をつけなければならないのは, 数値を指定しても文字列として渡されるということです。これは, プログラムで数値部分だけを取り出して数値に変換してやります。実数型ならval(), 整数型ならatoi()ですね。また, パラメータがなかったらヘルプメニューを出すというのも簡単にできるでしょう。

## ファイル処理

ツールとしてのBASICを見るとき, もっとも用途の広いものはファイル処理でしょう。X-BASICでは従来のBASICに比べ, 遙かに柔軟な処理が可能です。

テキストファイルの処理(けっこう面倒)やバイナリファイルの編集(簡単), 用途はグラフィックや音楽からAD PCM, マシン語へのパッチ当て(Oh!Xではよくやっている)といった分野にまで広がります。

基本的にファイルは連続したものとして扱われます。カセットテープがあってヘッド(ファイルポインタといいます)の位置を動かしてデータをアクセスするようなものです。ただカセットと違うのは, どれだけの距離を移動してもアクセス速度が変わらないということでしょう。どんな位置のデータでも手間は変わりません(フィールドなどといった面倒なものはない)。

また, 配列のようにそれぞれのデータにはアドレスが割り付けられていると見てもかまいません。配列と違うのは, あるデータを読み書きすると自動的に次のデータを処理する用意がされるということでしょうか。

X-BASICのファイル関係はC言語に似てシンプルな仕様になっているので, マニュアルのサンプルを見ていただければ, 使い方を特に解説するまでもないでしょう。ただ, ひとつだけ注意しておくことは「1文字入出力は大量のデータには使わないほうがいい」ということです。

これはX-BASICのfgetc(), fputc()が必要以上に遅いためです。大量のデータを扱うときは, ある程度まとめて配列に読み込んで処理したほうが圧倒的に速くなります。コンパイルしても遅いのでなるべくブロック単位で読み込んで処理するように心掛けましょう。

そのほか, テキストファイルを処理しようという場合には漢字の扱いに気をつけましょう(村田敏幸著『X68000マシン語プログラミング入門編』日本語の呪いの項を参照のこと)。X-BASICの文字列操作関数は半角文字を基本にしているので全角文字があると誤動作の原因になることがあります。

## さてさて

さて, 中東発尼寺直行便と呼ばれる某ゲームが発売され, 巷では阿鼻叫喚が響く今日この頃。プログラム開発には愛が大事だなと痛感します。特にエンディングはひどいものです。おそらくBASICのマニュアルしか見たことのない人でも, もっとスムースな処理が書けるでしょう(10行くらいかな?)。言語の特徴やハードウェアの特徴を知っておくことはプログラム技術以上に大事な場合だってありうるという教訓ですね。

BASICでだってC言語よりマシなことができるかもしれません。要は言語ではなく扱い方次第ということでしょう。ではBASICパーソナルプログラマの皆さんがんばってください。

グラフィックとファイル操作の基本

# カットファイルを作成しよう

Ishigami Tatsuya 石上 達也

VS2やSX-WINDOWなど，さまざまなところで表示することができるカットファイル。ここではどのようにしてカットファイルを作るのかを解説します。仕組みさえわかれば，簡単なグラフィックとファイル操作だけで実現できるものです。

1月号や5月号の付録ディスクを解凍しようとして驚いた方も多いのではないでしょうか？ 私がまず驚いたのは，絵のついた文書ファイルがVS.X上で表示できるということでした。SX関係の資料やサンプルも確かに貴重なものですが，文字だけの説明文に絵が入っていることの驚きは強烈でした。

## カットファイルの詳細をさぐる

さて，このような絵の出る.docファイルの仕掛けを見てみましょう。図1は5月号disk#1の¥quickstart¥vs2.docをコマンドラインからtypeしたものの一部です。

VS2.Xから表示させると絵の出るところに，なにやら%CUTとか書いてあります。この，

%CUT:−¥cut¥vs2.cut

というのが，たぶん，絵のデータが収まっているファイルを表しているのでしょう。そして続く，

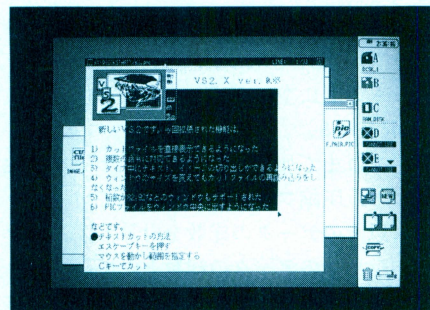
%CUT

というのが，きっと，絵の表示される領域を表しているのでしょう。と，だいたいの検討をつけたところで，詳細を探っていきます。

5月号のdisk#1をつらつらと眺めると，¥vs2¥VS_CUT.Sとか¥vs2¥VS_CUTLOAD.Sとかvs2¥VS_SAVE.Sとかが入っています。これらはアセンブラのソースファイルなので今回は解説しません。解析した結果のみを以下に挙げると，

1)　ファイルの先頭はASCIIコードで48バイトの文字列です。先頭は“CUT_V1.0”で，その後ろに（x,y）のように，絵の横，縦の大きさが数値ではなくASCII文字列で入ります。その後ろは，なんらかの文字列（注釈とかコメントとか）が入り，残りはスペース（ASCIIコードで$20_H$）で埋めます。45バイト目から$0D_H$,$0A_H$,$1A_H$が入りこのブロックが終わります。(だから，カットファイルをtypeすると,「CUTV-1.0(100,100)同心円だよーん」とか，出すことができる。うまく考えたもんだ)。

2)　2バイトずつX，Y方向の大きさが，今度は直接，数値で入ります。

3)　いよいよ画像データ。基本的にこのデータはラインごとの横方向データの集まりです。その心はというと，

3-1)　まず，このラインのデータの総数がまったりと入る。

3-2)　次に，データの圧縮状態を表すフラグがどどどどっと入る。

3-3)　最後に画像データがべたーと入る。ただし，このデータは上の列のデータとXOR　1　をとってある。

この画像データ列は横8ドット分の情報を1バイトにまとめてあります。モノクロデータは，1ドット分を1ビット（白か黒か）で表せるので，1バイト（＝8ビット）も使っていてはもったいない。そこで，8ドットを1バイトにまとめます。まとめ方は左方が最上位ビット，右側が，最下位ビットと決めておきます。ドットデータの数が，8の倍数でないときは，足りない分を黒のデータ（つまり0）があるとみなして，つじつまを合わせます。

VS2上のカットファイル

図1

```
%CUT:-¥cut¥vs2.cut
%CUT    ＶＳ２．Ｘ  ｖｅｒ．0.83
%CUT
%CUT
%CUT
%CUT
  新しいＶＳ２です。今回拡張された機能は，

1)  カットファイルを直接表示できるようになった
2)  複数の命令に対応できるようになった
3)  タイプ中にテキスト，イメージの切り出しができるようになった
4)  ウィンドウのサイズを変えてもカットファイルの再読み込みをしなくなった
5)  桁数が82,92などのウィンドウもサポートされた
6)  PICファイルをウィンドウ中央に出すようになった

などです。
●テキストカットの方法
  エスケープキーを押す
  マウスを動かし範囲を指定する
  Ｃキーでカット
●イメージカットの方法
  シフトキーを押しながらマウスを動かし範囲を指定
  Ｃキーでカット
```

## 基本方針

さて，このカットファイルを作るプログラムを作るのですが，まず最初に決めなければならないのは，どのように絵のデータを取り込むか？ ということでしょう。.pic→.cutコンバータや.gs3→.cutコンバータのようなものも考えられますが，ここではあっさりとこのプログラムが走り始める前にあらかじめグラフィック画面に表示されている絵を取り込むことにします（前述の方法はこれを読んでくれている方の課

題とします)。

具体的には,リスト1を見てください 20行から40行で,中心が(100,100)の同心円を10個表示しています。画像取り込み部を後ろにずらして,同心円の代わりにラムちゃんの絵を描いたってかまいません。

以上より,これから作るプログラムのおおざっぱな中身を決めましょう。

1) カットファイルのヘッダ部分3)を作る。
2) 取り込む領域を指定する。
3) 横,8ドット分の情報を1バイトにまとめる。
4) これから,1ラインごとにひとつ上のデータとXORを取る。いちばん上のラインは,その上に空白ラインがあると仮定してXORを取る。
5) これまで加工してきた画像データを眺めつつ,圧縮フラグの表を作っていく。
6) 以上のデータの総数を計算し,それをこのラインのデータの総数として,ファイルに出力する(実際はこのデータ自身の占める1バイト分もちゃんと計算に入れておく)。
7) 5)で作成した圧縮状態を表すフラグ2)の表をファイルに書き出していく。
8) 画像データのうち$00_H$でないもののみをファイルに書き出す。
9) 3)から8)までの処理を画像の縦の長さ分繰り返す。

注1) XOR(exclusive or 排他的論理和)論理演算のひとつなのだが,ここでは,その意味・背景などについては触れずにおく。作用だけを説明すると,

0 XOR 0 = 0
0 XOR 1 = 1
1 XOR 0 = 1
1 XOR 1 = 0←ここがorと違うところ

となる。

注2) 圧縮状態を表すフラグ

picなどの画像圧縮ファイルには驚くほど高度な技術が用いられていたが,ことcutファイルに関してはあまり複雑ではない。

eor演算を施したあとの画像データ1バイトを圧縮フラグ1ビットに対応させる。画像データが0(つまり8ドットまるまる空白)だったら,フラグは0,それ以外なら(つまり,8ドットのうちにひとつでも点があったなら)フラグは1とする。

例) 00000000 00@0@0@0 @0@0@@@@ @@@

という絵(?)があった場合,画像データは,

$00_H$, $2A_H$, $AF_H$, $E0_H$

であるから,圧縮フラグは$70_H$(2進数で0111 $0000_B$)となる。実際に出力するファイルは,

48バイトのASCII文字列
$1B_H$(横幅)
$01_H$(縦幅)
$05_H$(このラインの総数)
$70_H$(圧縮フラグ)
$2A_H$, $AF_H$, $E0_H$(実際の画像データ)

となる。

注3) ヘッダ

ファイルの先頭(ヘッド)部分のこと。同様に終わりの部分をフッダ(ヘッドに対してフット)と和製英語で呼ぶことがある。

## プログラムの詳細

今回の特集は,BASICのプログラミングについて論じるのであって,BASICのプログラム集ではないと思いますので,一般記事のプログラムの解説とは多少異なった視点で説明していきます。

まず,ヘッダ部分を作成する部分から作ってしまいましょう。250行からのmakeHeader関数がそれです。390行は文字列変数headerの中身が45文字未満だったら,残りに空白を足して45文字にしています。あとは簡単なので各自で解析してください。

先にも述べたように,絵のデータは8ドットで1バイトにまとめて扱います。だからといってpoint関数でちまちまやっていくのでは面倒です。ということで,8ドットまとめて取り込んでくれるpoint関数のようなものがほしくなります。そこで,920行からのgetGraph関数です。引数にX,Y座標を与えるとそこから右に8ドットまでの分のデータを1バイトにまとめて返してくれます。

次に,その取り込んだデータを圧縮する関数compressを作ります。この関数の中にはfor〜nextループが2つ入れ子になっています。内側のループで画像データ8バイト分の処理をしたあと,その外で圧縮フラグ(1バイト)の作成を行うという作業を繰り返しているのです。ここでは,これ以上の説明をするとかえってわかりにくくなるので,詳しく知りたい方は「カットファイルの詳細」の項とあわせて,自分で考えてみてください。理解の努力と引き替えに,いくらかの実力向上を約束します。

で,以上の処理を縦の幅分だけ,繰り返してやるわけです。これらの仕事は関数gcutが行います。540〜560行で配列変数buffをすべて0で初期化しているのは「プログラムの予備設計」の4)のところで述べた,いちばん上の行にあたる処理です。そのあとは圧縮後の画像データの大きさによって処理が分かれています。画像データがないときは,670行にいって,あっさり1(画像データもなし,圧縮フラグもなし,よって自分自身の大きさのみで1)をファイルに書き込んで終わりです。

画像データがあるときは,600行からこのラインのサイズ,圧縮フラグ,画像データとファイルに書き出しています。参考までに600行の数字の内訳は,

| | |
|---|---|
| 1 | 自分自身の大きさ |
| (x1−x0)/64+1 | 圧縮フラグの大きさ |
| size | 画像データの大きさ |

です。

注意深い人は,ここで,610行と640行に目が止まるでしょう。それぞれ,ループの終わりを表す数値がひとつ違うのです。これはこういうことです。配列変数の数がαなら,最後の変数はvar(α−1)で,最後の配列変数がvar(α)なら,配列の数はα+1。すべては,配列変数の初めの要素はvar(0)であり,我々の数の概念は1から始まることに起因しているのです。

## サンプルの使い方

プログラムの入力・理解はできたでしょうか? コンピュータ言語なんて,あれこれ考えずにとにかく自分の手で打ち込んで

リスト1

```
10 screen 2,0,1,1
20 for i=1 to 10
30 circle(100,100,i*5,7)
40 next
50 /*
60 /*  配列
70 /*
80 dim char cond(16)  /*圧縮状態を表すフラグのバッファ
90 dim char buff(128) /*生の画像データが入っているバッファ
100 dim char data(128) /*圧縮した画像データが入っているバッファ
110 /*
120 /*  グローバル変数
130 /*
140 int x0, y0, x1, y1 /*領域指定用
150 int cutFile        /*ファイルポインタ
160 str header[48]     /*ヘッダ部分を格納する
170 /*
180     makeHeader()
190     gcut()
200     fclose(cutFile)
210 end
220 /*
230 **ヘッダ部分を作る
```

サンプルの実行例

みることが上達の第一歩だと思います。ぜひ，自分の手で打ち込んでみてください。

このプログラムはrunすると，画面左端に紫色の同心円を描いていくつかの質問をしてきます。ここでは，とりあえず，以下のように答えます。

●“タイトルを入力してください。”
　CUT_V1.0（大文字で）
●“コメントを入力してください。”
　同心円だよーん
●“ファイルネームを指定してください。”
　test.cut
●“左上の座標　(x,y)”
　50,50
●“右下の座標　(x,y)”
　151,151

しばらくして，ディスクのアクセスランプが赤くなって，なにやらデータの書き込みを行います。アクセスランプが緑になって書き込みが終了したら，filesでtest.cutというファイルが作成されたのを確認してください。ちなみに，大きさは1096バイトでした。このファイルをタイプして，

　CUT_V1.0 (101,101) 同心円～

と表示されることを確認しましょう。

では，実際に表示させてみましょう。5月号の付録を解凍してできるDisk#1を書き込み可能な状態で用意します（バックアップを別に取っておいたほうが安全だろう）。ディレクトリ¥quickstartにList2を，¥cutに，先ほど作成されたtest.cutを入れておきます。そして，Disk#1を0ドライブに入れ，おもむろにX68000を起動します。起動が完了したら，test.docのところをダブルクリックしてみましょう。

SX-WINDOWを持っている方は5月号のリソースの拡張を行ったあと，test.cutをダブルクリックしてみてください。sximage.xのウィンドウの中で同心円が表示されているはずです。

それではこのサンプルを参考にして皆さんもカットファイルの表示に挑戦してみてください。

```
 240 */
 250 func makeHeader()
 260 str title
 270 str comment
 280 str fileName
 290     input"タイトルを入力して下さい。";title
 300     input"コメントを入力して下さい。";comment
 310     input"ファイルネームを指定して下さい。";fileName
 320     repeat
 330        input"左上の座標  (x,y) ",x0,y0
 340        input"右下の座標  (x,y) ",x1,y1
 350     until ( x0 < x1 and y0 < y1)
 360     cutFile = fopen( fileName, "c")
 370     header = title +"("+itoa(x1-x0) +","+itoa(y1-y0)+")"
 380     header = header + left$(comment, 26)
 390     if(strlen(header) < 45) then header = header + space$(4
5 - strlen(header))
 400     header = header+chr$(13)+chr$(10)+chr$(&H1A)
 410     fwrites(header,cutFile)
 420 /*横と縦の大きさを書き出す。
 430     fputc((x1-x0) /   &H100, cutFile) /*画像データの幅(横)
 440     fputc((x1-x0) and &HFF , cutFile)
 450     fputc((y1-y0) /   &H100, cutFile) /*画像データの幅(縦)
 460     fputc((y1-y0) and &HFF , cutFile)
 470 endfunc
 480 /*
 490 /*  実際に取り込む
 500 /*
 510 func gcut()
 520 int size
 530 /*
 540     for i=0 to (x1-x0)/8+1           /*配列の初期化
 550         buff(i) = 0
 560     next
 570     for y = y0 to y1
 580         size = compress()
 590         if(size <>0) then {
 600          fputc(2 + (x1-x0)/64 + size,cutFile) /*1行のサイズ
 610          for i = 0 to (x1-x0)/64
 620            fputc(cond(i), cutFile)
 630          next
 640          for i = 0 to size - 1
 650             fputc(data(i), cutFile)
 660          next
 670         } else fputc(1,cutFile)       /*1行全てが空白の時
 680     next
 690 endfunc
 700 /*
 710 /*  圧縮する
 720 /*
 730 func compress()
 740 int bt, flag
 750 int size
 760     size = 0
 770     for i = 0 to (x1 - x0) / 64
 780         flag = 0
 790         for bt=0 to 7
 800             p = getGraph(i*8+bt, y)
 810             if (p xor buff(i*8+bt))then {
 820                 data(size) = buff(i*8+bt) xor p
 830                 size = size + 1
 840                 flag = flag or (&H80 shr bt)
 850             }
 860             buff(i*8+bt)= p
 870         next
 880         cond(i) = flag
 890     next
 900     return(size)
 910 endfunc
 920 /*
 930 /*グラフィック画面を8ドット単位で取り込む
 940 /*
 950 func getGraph(x;int,y;int)
 960 int b,dx
 970     b = 0
 980     for dx = 0 to 7
 990         if(x0+x*8+dx > x1) then return(b)
1000         if(point(x0+x*8+dx,y)) then {
1010             b = b or (&H80 shr dx)
1020         }
1030     next
1040     return(b)
1050 endfunc
```

楽譜表示に挑戦

# MMLを画面に表示する

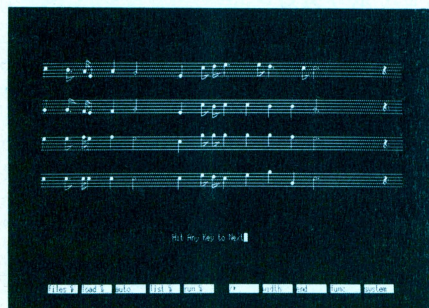
Ishigami Tatsuya　石上 達也

パソコンによる音楽というのも定着してきました。しかし，MMLを間違いなく入力するというのは難しいものです。見てわかりやすい五線譜のかたちに簡単に表示できたら……，とりあえずBASICで実験してみましょう。

Oh!X Live in '91などに載っている音楽プログラムを入力したことのある方ならわかると思いますが，ああいうMMLのプログラムって，間違いがあってもエラーは出ませんし，少しぐらいずれていてもなかなか気がつきません。なかにはエコー効果を出すため，わざと2声をずらすようなテクニックもあったりしてデバッグに手間取ってしまいます。こんなとき，MMLが視覚化できたらなぁとか思ってしまいます（32分音符ずれていても，なかなかわかりませんが，32ドットずれれば私にもわかります）。そこで，編集部からの「短くて，実用的で，しかも見た目が派手なプログラムを」という要請とからめて，今回はMMLを視覚化するプログラムを制作することにします。

## 方針

基本的に暇プロの一種であるので，あまり高度な機能はつけません。間違っても，MUSIC PRO-68Kのようなものは狙いません（でも，拡張性は持たせておきましょう）。とりあえず，音符と休符の表示ができればよしとします。

で，表示する量ですが4分の4拍子でいう4小節分あれば十分でしょう。拍子はMMLでは特に意味を持つものではないので無視します。つまり4分の4拍子で書かれたMMLであろうと，8分の6拍子で書かれたMMLであろうと，小節線を引かずにすべて同様に表示します。

チャンネルはX68000の場合8個あるのですが，画面に8セットも五線譜を引いていたのでは目がチカチカしてしまいます（バンドやオーケストラの楽譜を見たことがあるでしょうか？）。そこで，画面に表示するのは4セットの五線譜とし，ひとつの欄に2パートの楽譜を書き込んでいくことにします（混声合唱の楽譜がこんな感じです）。

音符はスプライトで表示したいような気もするのですが，横1列にたくさんの音符が並ぶ可能性のある今回のプログラムでは見送り，グラフィック画面に描いていくことにします。よって，スプライトも今回は使わないことだし，画面のモードは一番多くの情報を表示できるように，768×512ドットのモードで使うことにします。

で，横に4分の4拍子でいう4小節分の音符を表示するのですから，1小節あたり768/4＝192ドットということになります。1小節には最大で64分音符が64個入りますから，64分音符が横幅，192/64＝3ドット占めることにします。そうすると，自動的に32分音符が横幅6ドット占めることなります。以下同様に全音符が192ドット占めるというところまで決まっていきます（全音符が192ドット占めるということは，最初には決まりません。なぜなら64分音符が3ドットという区切りのよい値をとるかどうかは，そのときはまだわかりませんから）。

ちなみに，この，すべての音符の占める横幅が3の倍数であるということは付点の処理にたいへん都合がよいのです（たとえば，C..の占める横幅は48＊1.5＊1.5＝108で，きっちりと整数に収まります）。

あとは実際にコーディングしながら問題があったときに考えていきましょう。

と，かなりいきあたりで仕様を決めましたが，プログラミングというのはこんなもんです。キーボードに向かって，がつがつコーディングする前に，だいたいの仕様を決めます。コーディングという作業は，参考書などを覚えてしまえば，ある程度こなせるようになりますが，仕様決定というのは自分で実際にコツコツと実践を積み重ねていかなくては力はつきません。

逆に，目標に対して，つらつらつらぁーと仕様を煮詰めていくことができるようになったら，一人前といえます。

知恵と知識という言葉がありますが，知識にあたるのがコーディングで，知恵にあたるのが仕様決定です。今回の特集ではマニュアルに書いてあるような知識ではなくしっかりと知恵をつかみ取ってください。

## さらに仕様を煮詰める

ここでは，さらに深く仕様を考えていきます。まず，どのタイミングでMMLのデータを受け取るか決めます。X-BASICの場合，文字変数が255文字までという制約があるので，本物のOPMドライバのようにバッファに貯めておいていきなり吐き出すといった芸当はできそうにありません。もっとも適当そうなものとして，バッファ番号＝チャンネル番号という約束事を作って，バッファに貯め込むはずの瞬間に，そのデータを画面に表示することにします。

MMLへの対応は，行の先頭が，

(tn)　　(nは1から8までの整数)

で始まるものに注目，その他は無視します。今回サポートするMMLは次の6種類です。

●An～Gn　音符
　A～G：音程
　　後ろに＋または#を付けるとシャープ，－を付けるとフラット
　n：　音長(1～64，省略可)
●Rn　休符(1～64，省略可)
●.　符点
●L　デフォルトの音長（初期値＝4)
●On　オクターブ（初期値＝3）
●<　>　1オクターブ上/下

オクターブ指定の初期値が3になっていてX-BASICのMMLと違いますが，これは

五線譜の下に3オクターブ分，五線譜の上に3オクターブ分，五線譜の中に2オクターブ分を持ってきて上下対称に配置しようとしたからです。普通，なにも考えずにドといったら，下第1線上にあるような気がしますが，OPMDRVはその1オクターブ上の位置にとってしまうのです。

次に，どのタイミングで画面を切り替えるかを決めます。前述のように画面には4小節分しか表示できません。なにも考えずにどんどん表示していったのでは，画面が真っ白になってしまいます。よって，なんらかのタイミングで画面を初期化する必要があるのです。多くの場合，MMLでいちばん最初にくる音符のデータはトラック番号1のデータですから，このトラック番号1のデータが送られてきたら，画面を初期化することにします（440行）。

これで準備は万全のように思えますが，一時停止の機能がないと音符がだぁーっと表示されて，人間にはなにがなんだかわからなくなります。一時停止の機能としてESCキーによるものとmoreタイプのものをつけておきます。多くの場合，トラック番号8のデータが最後にくると思いますので，このデータを表示し終わった時点で，なんらかのキーが押されるのを待つことにします。

## いざコーディング

音符の絵のデータは莫大なものになりそうなので，線分と円弧とを組み合わせて作ります。2310行からが，それです。引数x,y,r はそれぞれ音符の中心位置のＸＹ座標，ひっくり返すかどうかのフラグ（1で返さない，－1で返す）です。

音符の中心とは，全体の重心などのことではなく，ここでは玉（図1参照）の中心の座標です。また，音符をひっくり返すというのは，音符が五線譜の上のほうにいったとき，見やすくするためのものです。上のパートの音符をそのままにして，下のパートの音符を，ひっくり返そうかとも思うのですが（合唱ではそうなっているので），X68000の画面では，いまいち見づらいのでやめておきましょう。

4分休符だけは円弧と線分だけでは作れそうにないので，例外的にPUT文用のデータを作って（110行―320行），それを表示しています。ここの部分のリストをよーく見てください。見ながら，目を細めて，だんだんOh!Xを離していきます。そうすると，なにかの模様になってきたでしょう。そう，4分休符のパターンが気に入らなかった場合の対処法はもうわかりましたね。

4分音符などの玉の塗り潰しを伴うところの処理を見てください。paint文が2つありますね。これは，X-BASICではPAINT文の境界色の指定ができないため，図2のような位置に玉がくると塗り潰しが行われません。そこで，罫線のちょっとずつ上と下から，玉を塗り潰してやるのです。

8分音符などの旗（図1参照）はCIRCLE文をたくさん使って描いています。この円弧のパラメータは，計算で求めたというよりも，実際に「えいやっ！」と，何種類か描いてみて都合のよい数字を選びました。こういう，やってみないと決められないパラメータの決定というのはBASICの得意な分野ですね。

ここらへんは，同じような文がいっぱい

**リスト**

```
 10 /*
 20 /*ＭＭＬを音符化して画面に表示するプログラム
 30 /*    Programmed by 石上　達也
 40 /*            '91 May 1st
 50 /*
 60 /*デフォルトの長さ
 70 dim defSpan(8) = {4,4,4,4,4,4,4,4,4 }
 80 /*デフォルトのオクターブ
 90 dim oct(8)     = {3,3,3,3,3,3,3,3,3 }
100 /*四分休符の記号
110 dim int r4chr(19) = {
120   &HF00000,
130    &HF0000,
140    &HFF000,
150    &HFFFF0,
160 &H777FFFFF,
170     &HFFFF,
180     &HFFFF,
190   &HFFFFFF,
200   &HFFFF00,
210  &HFFFFF00,
220 &H7FFFF777,
230    &HFF000,
240  &HFFFFFF0,
250 &HFF000FF0,
260 &HFF0000FF,
270 &HFF000000,
280 &HFFF77777,
290   &HFF0000,
300    &HF0000,
310     &HF000
320 }
330 /*
340 /*メインルーチン
350 /*
360 char ch
370 int fp,trkNo,linno
380 str mml[255],fn
390 linno = 1
400 input "ファイルネームを入力して下さい";fn
410 fp = fopen(fn,"r")
420 while(feof(fp) =0)
430     freads(mml,fp)
440     if(mml[0] <>'(') then continue
450     if(toupper(mml[1]) <>'T') then continue
460     if(mml[2] < '1' or '8' < mml[2]) then continue
470     if(mml[3] <>')') then continue
480         trkNo = mml[2]- '0'
490         if(trkNo = 1) then minit()
500         mtrk(trkNo,mid$(mml,5,strlen(mml)-3))
510         if(trkNo = 8) then {
520             locate 35,25:print"Hit Any Key to Next";
530             asc(inkey$)             /*一文字押されるまで待つ
540         }
550 endwhile
560 fclose(fp)
570 end
580 /*
590 /*五線譜を描く
600 /*
610 func minit()
620         screen 2,0,1,1
630         for j=1 to 4
640                 for i=1 to 5
650                     line(0,i*6+70*j,770,i*6+70*j,7)
660                 next
670         next
680 endfunc
690 /*
700 /*  ここにＭＭＬデータをわたす
710 /*
720 func mtrk(trkNo;int,mml;str)
730 int ptr,x,span
740         trkNo = trkNo-1
750         ptr = 0
760         x = 6
770 /*      メインループ
780         while(ptr < strlen(mml))
790 /*        現在注目している文字
800           ch = toupper(mml[ptr])
810           ptr = ptr + 1
820 /*        音を出す
```

ありますので，エディタが使える人はBASICから打ち込むよりは，そちらから打ち込んだほうが楽ではないでしょうか？

64分音符のパターンはあまり使わないうえに，無理して画面に表示させようとすると，かえって，汚らしくなってしまいます。そこで，32分音符のパターンと共用することにします。ただし，五線譜上に占める横の幅はちゃんと計算します。

これらの関数は，一応関数のかたちをとってはいますが，内容は音符のパターンを画面に描くだけです。これでは少々使いづらいでしょうから，これらの関数とメインの関数の間にワンクッション置いてやります。これが，関数onpです。引数はチャンネル数，X座標，音程，長さ，符点の数です。この関数の中で音符をひっくり返すとか，表示するY座標はどこだとか，どのパターンを表示するのか，といったことを決めてやります。

この関数を作ったことによって，メインの関数からはこれらのことは気にせずにすむことになります。

パターンの表示ができたところで，次に五線譜を画面に表示します。これも，カット＆トライで細かい数値を決めました。内容は見ればわかると思いますが，内側のループで罫線を1本ずつ描いていって，外側のループで五線譜を4セット描いています。

と，画面関係の下準備ができたところで，いよいよ，このプログラムの中心部を作ります。この関数は，music.fncのm_trk関数とコンパチにしておいて，

m_trk(trkNo,mml)

というかたちで呼び出すことにします。まず，trkNoですが，チャンネル数をOPMドライバでは1から数えるので，ここでは0

**図1　音符の各部の名称**

旗
棒
玉

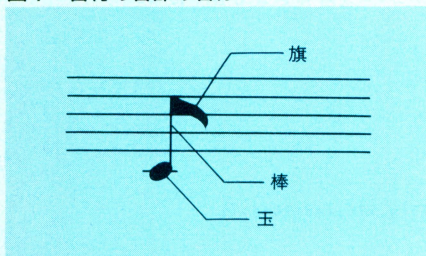



**図2　玉内部の塗り潰しができない例**

これを塗り潰そうとして
このようになってはいけない

```
 830            if ('A' <= ch and ch <='G') then {
 840                  if(ch <= 'B') then { feq = oct(trkNo)*7+ch
-'A'+5
 850                              } else { feq = oct(trkNo)*7+ch
-'C' }
 860 /*               臨時記号
 870                  ch = toupper(mml[ptr])
 880                  if(ch='+' or ch='#' or ch='-') then  {
 890                          y = (trkNo/2+1)*70+99 - feq * 3
 900                          if(ch = '-') then { flat(x-10,y)
 910                                     } else { sharp(x-10,y)}
 920                          ptr = ptr + 1
 930                          ch = mml[ptr]
 940                  }
 950 /*               音の長さ
 960                  if(isdigit(mml[ptr])) then {
 970                       span  = 0
 980                       while(isdigit(mml[ptr]))
 990                               ch = mml[ptr]
1000                               ptr = ptr + 1
1010                               span = span*10 + ch - '0'
1020                       endwhile
1030                  } else {
1040                          span = defSpan(trkNo)
1050                  }
1060 /*               付点の処理
1070                  futen = 0
1080                  while(mml[ptr] = '.')
1090                                  ptr = ptr + 1
1100                                  futen = futen + 1
1110                  endwhile
1120                  onp(trkNo,x,feq,span,futen)
1130                  x = x + 192 / span * pow(1.5#,futen)
1140            } else if(ch = 'R') then {          /*休符の処理
1150 /*               音の長さ
1160                  if(isdigit(mml[ptr])) then {
1170                       span = 0
1180                       while(isdigit(mml[ptr]))
1190                               ch = mml[ptr]
1200                               ptr = ptr + 1
1210                               span = span*10 + ch - '0'
1220                       endwhile
1230                  } else {
1240                          span = defSpan(trkNo)
1250                  }
1260 /*                 付点の処理
1270                  futen = 0
1280                  while(mml[ptr] = '.')
1290                          ptr = ptr + 1
1300                          futen = futen + 1
1310                  endwhile
1320                  rest(trkNo,x,span,futen)
1330                  x = x + 192 / span * pow(1.5#,futen)
1340            } else if(ch = 'L') then {            /*長さの指定
1350                  span = 0
1360                  while(isdigit(mml[ptr]))
1370                          ch = mml[ptr]
1380                          ptr = ptr + 1
1390                          span = span*10 + ch - '0'
1400                  endwhile
1410                  defSpan(trkNo) = span
1420            } else if(ch = 'O' and isdigit(mml[ptr])) then {
1430                       oct(trkNo) = mml[ptr] - '0'
1440                       ptr=ptr+1
1450            } else if(ch = '<') then {
1460                  oct(trkNo) = oct(trkNo) + 1
1470                  if(oct(trkNo) > 8) then oct(trkNo) = 8
1480            } else if(ch = '>') then {
1490                  oct(trkNo) = oct(trkNo) - 1
1500                  if(oct(trkNo) < 0) then oct(trkNo) = 0
1510 /*         無視するコマンド
1520            } else if(ch='Q' or ch='V' or ch='@' or ch='T' or
ch='&' or ch=' 'or ch='P') then {
1530                  while(isdigit(mml[ptr]))
1540                         ch = mml[ptr]
1550                         ptr = ptr + 1
1560                  endwhile
1570            } else {
1580                  print mml
1590                  print"左から";ptr;"番目の文字がエラーだよ"
1600            }
1610            endwhile
1620 endfunc
1630 /*
1640 /* 音符を表示する
1650 /*
1660 func onp(trkNo;int,x;int,feq;int,span,futen;int)
1670 int y0,y,i,r
1680            y0 = (trkNo/2+1)*70+36+3*21 /* 一番下のドの位置
1690            y = y0 - feq * 3
1700 /*半分よりだったら、ひっくり返す。
1710            if feq < 27 then r=1 else r=-1
1720            if feq <= 21 then {     /*下の罫線を追加
1730                    y0 = (trkNo/2+1)*70 + 36
1740                    for i = 0 to (21-feq)/2
1750                      line(x-8,y0+i*6,x+8,y0+i*6,7)
1760                    next
1770            }
1780            if feq >= 33 then {     /*上の罫線を追加
1790                    y0 = (trkNo/2+1)*70
1800                    for i = 0 to (feq-33)/2
1810                      line(x-8,y0-i*6,x+8,y0-i*6,7)
```

から数えるように変更しています（740行）。なぜかというと，単に趣味の問題です。

この関数の中には，ポインタを用意して，そのポインタの指し示す文字を解読して処理をしていくことに決めます。するともうこの関数の雛形はできましたね。

```
ptr = 0
while(ptr < strlen(mml))
 switch(mml [ptr])
     case '?' :
    ptr=ptr+1
         :
       break
   case '?' :
    ptr=ptr+1
         :
         :
       break
 endswitch
 ptr=ptr+1
endwhile
```

です。実際にSWITCH文で処理を振り分けると，音程の指定（A～Gのアルファベット）をはじめ，似たような処理をいくつも行わなければなりませんので，SWITCH文の代わりにIF文を並べています（SWITCHでも処理できます）。

このような文字列を解読していくプログラム（アセンブラとか，コンパイラとか）において「ポインタはその指し示す文字が不要になったときに初めて進めることができる」というルールを私は作っています。私はこれをかなり厳しく自分に課していて，宗教の戒律のように守っているのですが，これとは別の流儀で「ポインタは常に現在処理している文字のひとつ先を指し示している」というルールを作っている人もいます。どちらでもよいのですが，両者を切り替えるようなことをすると，途端にプログラムは読みづらくなりますし，保守性も極端に悪くなっていきます。どうしても，文字を先読みしなければならないという場合もありますが，極力，ポインタは統一性を持って操作するようにしましょう。

## さらなる発展のために

このへんで，このプログラムは打ち止めにします。誰かこのあとを継いでさらに完成度の高いプログラムを作ってください。今回は手を抜きましたが，いちばん下の音を表示しようとすると下のパートに引っ掛かってしまいます。この場合，8va┈┈┆(こ

```
1820                    next
1830            }
1840            switch span
1850              case  1: onpu1(x,y,r) :break
1860              case  2: onpu2(x,y,r) :break
1870              case  4: onpu4(x,y,r) :break
1880              case  8: onpu8(x,y,r) :break
1890              case 16: onpu16(x,y,r) :break
1900              case 32: onpu32(x,y,r) :break
1910              case 64: onpu32(x,y,r) :break
1920              default:print"エラーだよーん！" :break
1930            endswitch
1940 /*付点を付ける
1950            while(futen)
1960                  if feq mod 2 then {
1970                      pset(x+7,y+3,15)
1980                      circle(x+7,y+3,1,15)
1990                  } else {
2000                      pset(x+7,y,15)
2010                      circle(x+7,y,1,15)
2020                  }
2030                  x=x+4
2040                  futen=futen-1
2050            endwhile
2060 endfunc
2070 /*
2080 /*  休符を表示する
2090 /*
2100 func rest(trkNo;int,x;int,span,futen;int)
2110 int y
2120            y = (trkNo/2+1)*70+6
2130            switch span
2140              case  1: r1(x,y) :break
2150              case  2: r2(x,y) :break
2160              case  4: r4(x,y) :break
2170              case  8: r8(x,y) :break
2180              case 16: r16(x,y):break
2190              case 32: r32(x,y):break
2200              case 64: r32(x,y):break
2210              default:print"エラーだよーん！" :break
2220            endswitch
2230 /*付点を付ける
2240            while(futen)
2250                    pset(x+13,y+15,15)
2260                    circle(x+13,y+15,1,15)
2270                    x=x+4
2280                    futen=futen-1
2290            endwhile
2300 endfunc
2310 /*
2320 /* 全音符
2330 /*
2340 func onpu1(x;int,y;int,r;int)
2350          circle(x,y,3,15)
2360 endfunc
2370 /*
2380 /* 二分音符
2390 /*
2400 func onpu2(x;int,y;int,r;int)
2410          circle(x,y,3,15)
2420          line(x+r*3,y,x+r*3,y-r*21,15)
2430 endfunc
2440 /*
2450 /* 四分音符
2460 /*
2470 func onpu4(x;int,y;int,r;int)
2480          circle(x,y,3,15)
2490          paint(x,y+1,15)
2500          paint(x,y-1,15)
2510          line(x+r*3,y,x+r*3,y-r*21,15)
2520 endfunc
2530 /*
2540 /* 八分音符
2550 /*
2560 func onpu8(x;int,y;int,r;int)
2570          circle(x,y,3,15)
2580          paint(x,y+1,15)
2590          paint(x,y-1,15)
2600          if(r=1) then {
2610                  line(x+3,y,x+3,y-21,15)
2620                  circle(x+3,y,21,15,45,90,320)
2630                  circle(x+3,y,30,15,60,90,128)
2640          } else {
2650                  line(x-3,y,x-3,y+21,15)
2660                  circle(x-3,y,21,15,270,315,320)
2670                  circle(x-3,y,30,15,270,300,128)
2680          }
2690 endfunc
2700 /*
2710 /* 十六分音符
2720 /*
2730 func onpu16(x;int,y;int,r;int)
2740          circle(x,y,3,15)
2750          paint(x,y+1,15)
2760          paint(x,y-1,15)
2770          if(r=1) then {
2780                  line(x+3,y,x+3,y-21,15)
2790                  circle(x+3,y,21,15,45,90,400)
2800                  circle(x+3,y,19,15,60,90,250)
2810                  circle(x+3,y,13,15,45,90,300)
2820                  circle(x+3,y,16,15,55,90,150)
2830          } else {
```

のところは，楽譜よりも１オクターブ下で演奏してくださいの意）と五線譜の下に書いて，ちゃんと，隣のパートに掛からないようにするべきでしょう。スラーとかタイなどもしっかりサポートすると，がぜん見栄えがよくなります（ヒント：２点を結ぶ半径ｒの円弧を求めるというのは，高校でさんざんやらされているでしょ？）。

また，楽譜には，連桁処理というのがあって，図３（ａ）のような場合は図３（ｂ）や（ｃ）のように記譜するのが普通です。どんなところで，連桁処理を施すかということは，けっこう奥の深そうな問題ですが，どなたか，チャレンジしてみてください。

繰り返し記号をサポートしていないというのはやはりさびしいものです。もちろん，繰り返し記号があったら，その部分を繰り返すわけではありません（バッファリングしていないので，そのようなことはできません）。「繰り返し記号がここにあったよ」と画面に表示してやればよいのです。読者の方にぜひやってほしい拡張のひとつです。

今回のプログラムをＣで書き直せば，文字列は255文字以内という制約がありませんし，スピードは（今回のプログラムに関していえば）十分です。ひょっとしたら，リアルタイムで楽譜を表示して，画面が一杯になったら，つつつっーと右スクロールするようなこともできるかもしれません。

また，表示の方法を，music.fncを抜いておいてミュージック関数を呼んだときにそのデータを横取りして表示するようにすれば，プログラムを打ち込みながら，デバッグできるようになるでしょう。

そして，最終的には，このプログラムをデバイスドライバの形で常駐させcopyコマンドなどで，MMLデータをopmの代わりにこのプログラムに渡してやると，音符を画面に表示するようすれば非常に面白いのではないでしょうか。

**図3　連桁処理**

(a)　(b)

このような音譜は　こうなったり

(c)

ときには，こうなったりします

```
2840                    line(x-3,y,x-3,y+21,15)
2850                    circle(x-3,y,21,15,270,315,400)
2860                    circle(x-3,y,19,15,270,300,250)
2870                    circle(x-3,y,13,15,270,315,300)
2880                    circle(x-3,y,16,15,270,305,150)
2890            }
2900 endfunc
2910 /*
2920 /* 三十二分音符
2930 /*
2940 func onpu32(x;int,y;int,r;int)
2950            circle(x,y,3,15)
2960            paint(x,y+1,15)
2970            paint(x,y-1,15)
2980            if(r=1) then {
2990                    line(x+3,y,x+3,y-r*21,15)
3000                    circle(x+3,y,21,15,45,90,400)
3010                    circle(x+3,y,19,15,60,90,250)
3020                    circle(x+3,y,15,15,50,90,300)
3030                    circle(x+3,y,21,15,63,90,150)
3040                    circle(x+3,y,12,15,45,90,180)
3050                    circle(x+3,y,15,15,60,90,100)
3060            } else {
3070                    line(x-3,y,x-3,y+21,15)
3080                    circle(x-3,y,21,15,270,315,400)
3090                    circle(x-3,y,19,15,270,300,250)
3100                    circle(x-3,y,15,15,270,310,300)
3110                    circle(x-3,y,21,15,270,297,150)
3120                    circle(x-3,y,12,15,270,315,180)
3130                    circle(x-3,y,15,15,270,300,100)
3140            }
3150 endfunc
3160 /*
3170 /* 全休符
3180 /*
3190 func r1(x;int,y;int)
3200            y = y / 70 * 70 +13 /*定位置だけに許す
3210            fill(x-3,y,x+3,y+3,15)
3220            line(x-7,y,x+7,y,15)
3230 endfunc
3240 /*
3250 /* 二分休符
3260 /*
3270 func r2(x;int,y;int)
3280            y = y / 70 * 70 +13 /*定位置だけに許す
3290            fill(x-3,y+3,x+3,y+6,15)
3300            line(x-7,y+6,x+7,y+6,15)
3310 endfunc
3320 /*
3330 /*  四分休符
3340 /*
3350 func r4(x;int,y;int)
3360            y = y / 70 * 70 +8 /*定位置だけに許す
3370            put(x,y,x+7,y+19,r4chr)
3380            circle(x+11,y+8,1,15)
3390            circle(x+11,y+13,1,15)
3400 endfunc
3410 /*
3420 /* 八分休符
3430 /*
3440 func r8(x;int,y;int)
3450            y = y / 70 * 70 +14/*定位置だけに許す
3460            circle(x,y,5,15,110,359)
3470            line(x+6,y,x,y+13,15)
3480 endfunc
3490 /*
3500 /* 十六分休符
3510 /*
3520 func r16(x;int,y;int)
3530            y = y / 70 * 70 +12/*定位置だけに許す
3540            circle(x,y,4,15,170,359)
3550            circle(x-2,y+4,4,15,170,359)
3560            line(x+6,y,x-1,y+15,15)
3570 endfunc
3580 /*
3590 /* 三十二分休符
3600 /*
3610 func r32(x;int,y;int)
3620            y = y / 70 * 70 +10/*定位置だけに許す
3630            circle(x,y,4,15,175,359)
3640            circle(x-1,y+3,4,15,175,359)
3650            circle(x-2,y+6,4,15,175,359)
3660            line(x+6,y,x,y+17,15)
3670 endfunc
3680 /*
3690 /*  嬰記号
3700 /*
3710 func sharp(x;int,y;int)
3720            line(x-5,y+4,x+5,y+1,15)
3730            line(x-5,y-1,x+5,y-4,15)
3740            line(x-2,y-6,x-2,y+7,15)
3750            line(x+2,y-7,x+2,y+6,15)
3760 endfunc
3770 /*
3780 /*  変記号
3790 /*
3800 func flat(x;int,y;int)
3810            line(x-2,y-8,x-2,y+3,15)
3820            circle(x-2,y,3,15,270,90)
3830 endfunc
```

使い慣れた言語でのツール作り

# スプライトを加工する

Hamazaki Masaya 浜崎 正哉

拡大/縮小/回転……，いずれも最近のゲームで流行っているものです。ハードウェアで処理しきれない部分をソフトウェアで補うことによりこういったものも可能になります。ここではスプライトエディタにありそうでない機能，回転の実現方法を探ってみましょう。

## BASICをどう使ってます？

ふと思いついたことや急いでちょっとしたツールを作らなくてはならなくなってしまったときに，皆さんは何を使ってプログラミングをしますか？　多くの人は，そのような場面に直面したら迷わずBASICと答えると思います。ひと通りのことができ，コストもかからず（なにせ買ったときから付いていますからね）プログラミングユーザーが初めて触れる言語で誰もが理解しているでしょうから。

しかし，開発のメインとしてBASICを使うのは少し無理があるかもしれません。なにしろBASICは実行速度が遅い。これは致命的といえます。そしてこの問題に直面した多くのユーザーはコンパイラ言語，アセンブラへと移行していくことでしょう（僕自身もそのひとり）。逆にいえばそれほど凝ったことをしなければBASICでも十分用が足りるともいえます。

そういうことで簡単で手間いらず，拡張しやすいツールを目指して何かプログラムを作ってみましょう。

## さあ，何をやろうか

といってもいきなり考えが浮かぶわけでもなく，ぼーっとしながら何を作ろうか考えていてふと思いついたのがスプライトの回転パターンを自動作成してくれるツールです。市販，もしくは付属のスプライトエディタにも回転機能がついていますがたいていは90度単位の回転のみしかないでしょう。回転を滑らかに行わせようとしたら，せめて45度単位くらいで必要だし，できれば任意角度での回転ぐらいしてほしいものです。

ひとつや2つの回転パターンを作成するだけなら自分でパターンを作成してもいいのですが，数が多くなるにつれパターンが大きくなるにつれてその作業は非常につらくなるでしょう。多少，変換後のデータは汚くてもある程度基本パターンを作ってくれるだけでもそうとうな労力が軽減されますからね。

それとすでに定義されているデータを加工するだけなら，時間も手間もかからずにBASICで簡単に実現できるでしょう。よし，決まり。

## 準備体操をしておこう

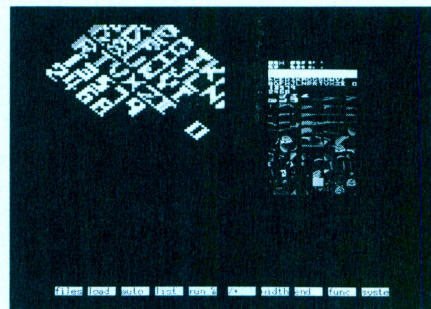
これで何を作るか目標ができました。が，プログラムを組む前にどういったプログラムを作るのか，もう少し考えてみることにします。どういうことかというと，仕様書を作るのです。確かにスプライトの回転パターンを自動作成してくれるツールを作ろう，そう思ったわけですが目標だけではプログラグラムは作れません。当たり前のことですがどのようにして作成していくか，どういったものが必要になるか，前もって紙の上または自分の頭の中に構成しておくと開発につまったとき非常に役に立つことがあります。

プログラムが予想以上に大きくなってしまったときなど，自分がどのようなルーチンを作っていたかメモしてあれば，途中で投げ出すこともなくメモを見ながら再検討もできるでしょう。途中でわけがわからなくなって投げ出してしまうのは初心者が陥りやすい点のひとつでもあります。面倒ですがプログラムを組む際の前準備はしっかりしておくことを僕はおすすめします。

さて話を戻しましょう。仕様書を作っていこうということでしたね。どのような機能を持たせるか先ほど話しましたが，もう一度，箇条書きで書いていきます。

・複数個のPCGを組み合わせたパターンに対応する

・任意の角度で回転させるようにする

・加工したデータの表示機能

スプライトツール完成版

以上の3つです。次に制限事項として，

・組み合わせパターンは，n×nの正方形のパターンのみ（PCG取り込みの簡略化のため）

・表示上の関係からn<=4とする

・組み合わせパターンのPCG番号は連続していなければならない（図1参照）

・加工するPCGデータは定義しておかなくてはならない

今度はこれらの機能を実現するために，必要と思われるサブルーチンを考えてみます。

1)　PCGデータを取り込む

2)　取り込んだデータを変換する（回転処理を加える）

3)　変換したデータをPCGに定義

4)　加工したデータを4倍にしてグラフィックに表示

5)　パレットブロックのパレットデータを

図1

| 1 | 2 | 3 |
|---|---|---|
| 4 | 5 | 6 |
| 7 | 8 | 9 |

48×48ドットのパターンの場合

読み込む

さらにワークエリアを考えていきます。

・int型，64×64個の２次元配列を２つ（PCG取り込み用と加工用作業エリア）

・char型，256個の１次元配列（PCGの一時取り込み用）

・int型，16個の１次元配列（パレットデータ保持用）

これでほぼ完璧でしょう。ついでにサブルーチンの副作用も前もって考えておくといいと思います。あとはアルゴリズムを考えてやるだけです。

## 目標に向けてプログラミング!

回転ルーチンはあと回しにして，まずはリスト１。これは画面初期化とメインルーチンのリストです。何をどのように初期化しているかを見ながら打ち込んでみてください。内容については説明するまでもないでしょう。わからないことがあればマニュアルを参照してください。

で，今度はPCGデータの取り込みルーチン（リスト２）にいきましょう。取り込み方法については図１を見てもらえばわかると思います。これは48×48ドットのパターンの場合で数字はPCG番号を表しています。スプライトのPCGデータは16×16をひとつの単位としていて，大きなパターンというのはそれらを組み合わせて作っていきます。つまり32×32ドットのパターンというのは２×２＝４つのPCGで構成されていることになります。リスト中，はじめの２重ループはPCGの縦と横の個数分で，次の２重ループでsp_pat( )で取り出したデータをワークに格納しています。このリストの中で１次元配列に取り出したPCGデータを２次元配列に転送しているところがポイントです。

こうして取り込んだデータを今度はグラフィックに表示させたいということで，まずはそのPCGに対応しているパレットデータを取り出します。リスト３のpallet_get( )という関数がそれです。なぜこのようにループを組んでやってパレットデータを取り出してやらなくてはならないかというと，パレットデータを取り出してくれる関数がないからです。しかもパレット設定用関数の副作用を使って定義しています。非常に泥臭い方法ですがほかに打つ手がないのです。４倍表示については取り込んだデータに相当するパレットコードでボックスフルを使ってドットを描いています。

以上，リスト１からリスト３まで打ち込んでとりあえず実行してみてください。すると２×２のサイズでPCG番号０からのパターンが画面に表示されると思います。もしもうまくいかなかった場合，最初に考えられる点はPCGデータが定義されていないというものでしょう。定義されていないデータを表示しようとしてもうまくいくはずがありませんね。このプログラムを実行する前に，とりあえずゲームを立ち上げておくとかスプライトエディットツールでパターンを制作しておきましょう。まあ，これは単なる自分自身の間違いですがエラーが起きずに異常な表示がされた場合は，エラー状況をよく把握してどのサブルーチンがおかしいのかつきとめなければなりません。それは，どのサブルーチンがどういった動作をするのか注釈を見ながらチェックしていきましょう。

## 回転するには?

いよいよ回転ルーチンの説明です。基本は高校の代数幾何の教科書に載っている１次変換式を使います。参考までにその式を書いてみると，

X′＝X×cos(a)−Y×sin(a)

Y′＝X×sin(a)＋Y×cos(a)

となっています。その式どおりプログラムを組んでみたのがリスト４で，試しにこのまま打ち込んで130行目の注釈を削除して540行を，

540　col＝pallet(work_sub(i,j)

に変更してから実行してみると，

添字の値が異常です

と，エラーが起きてしまいます。ちょっと考えると回転処理を加えたとき，図２のようになりますね。ところが変換後に斜線部分が配列のエリアからはみだしてしまいます。もちろんそこには配列が用意されているわけではないのでBASICは指定外の値を受け取ってエラーを出力してしまうのです。原因がわかれば対処のしようもあるというものです。要するに変換後の座標が指定外の数値になってしまったらそのデータ

リスト1

```
 10 int i,j,l,m
 20 dim int work(63,63)
 30 dim int work_sub(63,63)
 40 dim int pallet(15)
 50 dim char pat_get(255)
 60 /*
 70 /*メインルーチン
 80 /*
 90   scr_init()
100   pallet_get(1)
110 /*pcg_put()
120   work_get(2,0)
130 /*roll(45,2)
140   put_gra(2)
150   end
160 /*
170 /*画面の初期化
180 /*
190 func scr_init()
200   screen 1,3,1,1
210   wipe()
220   sp_clr(255)
230   bg_fill(0,255)
240   bg_set(0,0,1)
250   bg_scroll(0,0,0)
260   sp_off(0,127)
270   sp_disp(1)
280 endfunc
```

リスト2

```
290 /*
300 /*size×sizeのスプライトをワークに格納
310 /*   引数(パターンサイズ,読み込みPCGナンバー)
320 /*
330 func work_get(size,number)
340   for i=0 to size-1
350   for j=0 to size-1
360      sp_pat(number,pat_get,1)
370      number=number+1
380      for l=0 to 15
390      for m=0 to 15
400         work(m+16*i,l+16*j)=pat_get(m*16+l)
410      next
420      next
430   next
440   next
450 endfunc
```

リスト3

```
460 /*
470 /*取り込んだデータを4倍にしてグラフィックに表示
480 /*   引数(パターンサイズ)
490 /*
500 func put_gra(size)
510 int col
520    for i=0 to size*16-1
530    for j=0 to size*16-1
540       col=pallet(work(i,j))
550       fill(j*4,i*4,j*4+4,i*4+4,col)
560    next
570    next
580 endfunc
590 /*
600 /*パレットデータの取り出し
610 /*
620 func pallet_get(pal_num)
630   for i=0 to 15
640      pallet(i)=sp_color(i,0,pal_num)
650      sp_color(i,pallet(i),pal_num)
660   next
670 endfunc
```

は無視すればいいのです。そこで850行と870行を追加して,

850 if (x>=0 and y>=0) and(x<size*16 and y<size*16) then {

870 }

としておきましょう。これでエラーもなくなりとりあえず完成かな,と思いきやとんでもない落とし穴がありました。

リストを実行させてみると,何か表示がおかしいことに気づくと思います。試しに四角いベタ塗りのパターンで試してみるとそれがよくわかります。確かに変換したパターンは結構汚いものですがそれ以前にパターンに穴があいてしまうのはちょっと問題。いろいろアルゴリズムの変更を試みたりしてみましたがどうにもならなくて,さんざん悩んだ末,A.T.氏に泣きつくことにしました。A.T.氏いわく「そういうときは変換したほうのドットが元のパターンのどこから変換したか調べればいいんですよ」とのこと。僕が納得いかないような顔をしているとすかさず図を描いて説明され,ようやく納得しました。

まず,図3を見てください。図3-Aの実線部分を45度回転させるとBの実線部分になりパターンは点線部分に収まるわけです。逆に考えれば点線部分を−45度回転させたもののパターンを元のパターンから取り出してもかまわないわけです。具体的には760行と860行をそれぞれ,

760 rd=pi(rd)/180

860 work_sub(j,i)=work(x,y)

に変更して実行してみてください。不思議なことに穴はなくなっているでしょう。画像を変換したときには元の画像を加工してうまくいかなかった場合,加工されたものが元の画像のどこにあたるのか調べていくという逆の手順をふんでやるのが基本ということです。拡大縮小の操作も,拡大されたまたは縮小されたパターンのドットが元のパターンのどこからやってきたか調べるとうまくいくそうです。

修正前。穴だらけだ。

修正後。穴がふさがっている

ということでどうにか目標は達成できました。いやあ一時はどうなるかと思いましたよ。不完全なものを発表するわけにもいかず,いろいろあがいたけどうまくいかない。結局は人に頼ることになってしまいましたけれど。最後に訂正したリスト1からリスト4までまとめて掲載しておきます。作成したデータをPCGに定義するサブルーチンと定義されているPCGデータを画面の右に表示するサブルーチンも加えておきましたので参考にしてください。これらについては説明の必要もないでしょう。

## ツールとして完成させる?

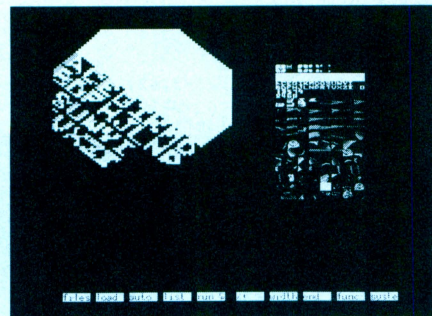
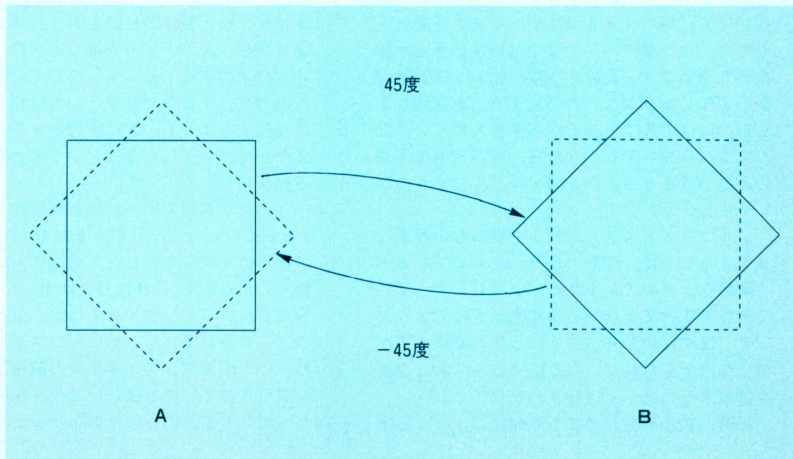
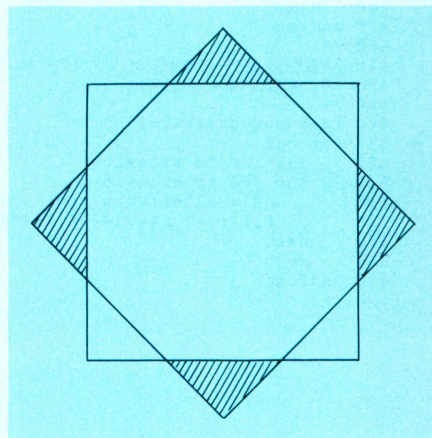
このリストは本当に骨格だけの情けないものですがさらに拡大縮小処理を付けたりパターンの指定をマウスで行うようにしたり,好きなだけ拡張してみてください。あまり欲張りすぎるとBASICでは使いものにならなくなってしまうと思いますので,ほどほどにしておいたほうがいいかもしれません。特に初心者の人たちには最初から気合いを入れて組むのはつらいものがあるでしょうから,こういった簡単なものを理解して自分なりに必要な機能を拡張してみるといい勉強になります。

今回,久しぶりにBASICなるものを触ってプログラムを組むことになったので,覚えているかどうかあまり自信がなかったのですがやっぱり体が覚えています。メインの開発言語が移ってBASICなどもう触ることなどないと思っていました。プログラムを組み終わり,こういった位置付けならBASICもあまり苦痛を覚えずに使うことができるんだな。そう思いました。BASICだけにかぎらず皆さんも必要に応じて,状況に応じて開発言語を使い分けてみましょう。

図2

リスト4

```
680 /*
690 /*回転ルーチン
700 /*   引数(回転角度,パターンサイズ)
710 /*
720 func roll(rad,size)
730 float dx,dy,rd,s,c
740 int x,y
750 rd=rad
760 rd=pi(-rd)/180
770 s=sin(rd)
780 c=cos(rd)
790   for i=0 to size*16-1
800   dy=i-(size*16/2-1)
810     for j=0 to size*16-1
820       dx=j-(size*16/2-1)
830       x=dx*c-dy*s+(size*16/2-1)
840       y=dx*s+dy*c+(size*16/2-1)
860                 work_sub(x,y)=work(j,i)
880     next
890   next
900 endfunc
```

図3

45度

−45度

A

B

リスト5

```
  10 int i,j,l,m
  20 dim int work(63,63)
  30 dim int work_sub(63,63)
  40 dim int pallet(15)
  50 dim char pat_get(255)
  60 /*
  70 /*メインルーチン
  80 /*
  90   scr_init()
 100   pallet_get(1)
 110   pcg_put()
 120   work_get(2,0)
 130   roll(45,2)
 140   put_gra(2)
 150   end
 160 /*
 170 /*画面の初期化
 180 /*
 190 func scr_init()
 200   screen 1,3,1,1
 210   wipe()
 220   sp_clr(255)
 230   bg_fill(0,255)
 240   bg_set(0,0,1)
 250   bg_scroll(0,0,0)
 260   sp_off(0,127)
 270   sp_disp(1)
 280 endfunc
 290 /*
 300 /*size×sizeのスプライトをワークに格納
 310 /*  引数(パターンサイズ,読み込みPCGナンバー)
 320 /*
 330 func work_get(size,number)
 340   for i=0 to size-1
 350   for j=0 to size-1
 360      sp_pat(number,pat_get,1)
 370      number=number+1
 380      for l=0 to 15
 390      for m=0 to 15
 400          work(m+16*i,l+16*j)=pat_get(m*16+l)
 410      next
 420      next
 430   next
 440   next
 450 endfunc
 460 /*
 470 /*取り込んだデータを4倍にしてグラフィックに表示
 480 /*  引数(パターンサイズ)
 490 /*
 500 func put_gra(size)
 510 int col
 520    for i=0 to size*16-1
 530    for j=0 to size*16-1
 540       col=pallet(work_sub(i,j))
 550       fill(j*4,i*4,j*4+4,i*4+4,col)
 560    next
 570    next
 580 endfunc
 590 /*
 600 /*パレットデータの取り出し
 610 /*
 620 func pallet_get(pal_num)
 630   for i=0 to 15
 640      pallet(i)=sp_color(i,0,pal_num)
 650      sp_color(i,pallet(i),pal_num)
 660   next
 670 endfunc
 680 /*
 690 /*回転ルーチン
 700 /*  引数(回転角度,パターンサイズ)
 710 /*
 720 func roll(rad,size)
 730 float dx,dy,rd,s,c
 740 int x,y
 750 rd=rad
 760 rd=pi(rd)/180
 770 s=sin(rd)
 780 c=cos(rd)
 790   for i=0 to size*16-1
 800   dy=i-(size*16/2-1)
 810     for j=0 to size*16-1
 820       dx=j-(size*16/2-1)
 830       x=dx*c-dy*s+(size*16/2-1)
 840       y=dx*s+dy*c+(size*16/2-1)
 850         if (x>=0 and y>=0)and(x<size*16-1 and y
<size*16-1) then {
 860                 work_sub(j,i)=work(x,y)
 870         }
 880     next
 890   next
 900 endfunc
 910 /*
 920 /*定義されているPCGをBGに表示
 930 /*
 940 func pcg_put()
 950   for i=0 to 15
 960   for j=0 to 7
 970      bg_put(0,j+20,i+4,i*8+j+256)
 980   next
 990   next
1000 endfunc
1010 /*
1020 /*加工したスプライトデータを定義する
1030 /*  引数(パターンサイズ,定義先PCGナンバー)
1040 /*
1050 func pcg_set(size,number)
1060   for i=0 to size-1
1070   for j=0 to size-1
1080      for l=0 to 15
1090      for m=0 to 15
1100         pat_get(m*16+l)=work_sub(m+16*i,l+16*j)
1110      next
1120      next
1130      sp_def(number,pat_get,1)
1140      number=number+1
1150   next
1160   next
1170 endfunc
```

## 高速化と愛のBASIC

最後にもう少しだけプログラムについて触れておきます。まずこのプログラムの実行速度は遅いです。特に4×4個のキャラクタを回転させると，ひと眠り……とまではいきませんが結構待たされます。BASICの苦手なループ回数が多いのでしょうがないといえます。高速化をするためにはアルゴリズムから手を入れていかなくてはならないでしょうから，実行速度に不満のある方はさっさとコンパイルしてしまうのがいいでしょう。

アルゴリズムに手を入れる必要のない簡単な高速化としては，関数の細分化をやめてなるべく関数をまとめてしまうといいでしょう。そして引数をすべてグローバル変数にしてしまうのです。確かにわかりづらくなってしまいますが背に腹は代えられない，ということで実行速度を追求する人はがんばりましょう。

一時，表示部分だけでも高速化しようとしましたがよけいな条件判断文が入ったため，かえって遅くなるという非常にむなしい結果に終わりました。拡大表示エリアをボックスフルで黒く塗り潰し，黒のドットを描くときには塗り潰ぶさずに次のドットにスキップしていくようにしたのです。インタプリタという性格上，1行を最適化するよりもいかに実行させる行数を減らすかがポイントになる，ということを思い知らされました。

なんだかんだでいいわけと実行速度が遅い！とBASICの文句ばかり書きましたがプログラムの制作は非常に楽でした。放っておいても手が動くし足も動く(これは貧乏ゆすり)。快適な気分でプログラミングできました。最近はプログラミングをしているとだいたいストレスが溜まっていくのです。特にメインの開発言語がアセンブラなのでいつも楽しいエラーを拝ませてもらっています。いちばん楽しかったのは，

　アドレスエラーが発生しました
　バスエラーが表生しました
　おかしな命令を実行しました

と順番にエラーが表示されて最後に，

　アバスエラーが発生しました

と表示されたときでした。そのときはさすがにいらつく気力も失せて，「X68000っておちゃめだなあ」と感心しました。

BASICではこういったことが起きませんし，エラーが発生してもプログラムが思いどおりに動かなくても「かあいいヤツ」と，なんか微笑ましい気分になれるのです。なにしろパソコンとのつきあいが，BASICのプログラムを入力することから始まりましたからね。僕にとっては特に思い出深い言語です。皆さんも愛を持ってBASICと接してみるのもいいと思います。でもプラトニックに止めておかないとアブナイ人と思われてしまいますから注意しましょう。

外部関数の作成

# X-BASICでMAGICを

Kageyama Hiroaki 影山 裕昭

必要に応じてBASICを拡張する方法が明らかにされているのもX-BASICの特徴です。ここではMAGICをX-BASICから使えるようにしてみましょう。手軽に実行できますので，MAGIC用ユーティリティ作成などに役立つでしょう。

3D表示のゲームが作りたい。高速グラフィックの世界を楽しみたい。誰でもそんな思いを持ったことがあると思います。MAGICのX68000版（5月号の付録ディスクに収録）はまだまだ高速というには恥ずかしい段階ではありますが，SIONを見てもらえばMAGICの魅力の一端がわかってもらえたと思います。遊んでいて自分でもMAGICを使ってみたいと思った読者もたくさんいるでしょう。でも，アセンブラがわからないからダメだ，と諦めてしまった人も多いんじゃないでしょうか？

そこで，今月はMAGICをX-BASICから使えるようにするプログラム，MAGIC.FNCを発表します。X68000を買えばついてくるX-BASICでMAGICが使えるようになれば，アセンブラやリンカを持っていなくてもプログラムを書くことができます。6月号の読者アンケートの結果からも，約半数の人が開発に使っている言語としてBASICを挙げています。同じ質問にアセンブラと答えた人はわずか15%。これでMAGICを使ったアプリケーション作成に参加しやすい環境が整いました。

## 入力および組み込み方法

リスト1のソースリストを入力します。アセンブルには付録ディスクに収録したMAGIC.H，MAGIC.MACの2つのインクルードファイルのほかに，Cコンパイラに付属のインクルードファイルが必要です。Cコンパイラを使っている方で，すでに環境変数includeが設定されているなら，includeで指定されるパスにMAGIC.HとMAGIC.MACをコピーしておけばアセンブルできるはずです。そうしてアセンブル，リンクしてできた実行ファイルをMAGIC.FNCに変更したら，BASIC.CNFに，

　FUNC＝MAGIC

を加えて，BASIC.CNFと同じディレクトリにMAGIC.FNCをコピーしておきます。

なお，MAGIC.FNCは付録ディスクに収録したMAGIC.Xのコマンドでグラフィック表示をします。そんなわけで，MAGIC.FNCを使うには，あらかじめMAGIC.Xを常駐させておかないとエラーが発生します。コマンドラインから，

　A>MAGIC

でMAGICを常駐させてください。

MAGIC.FNCの作成にあわせてMAGIC.XのコマンドMODEとDISPLAY 2Dを改良しました。リスト2をMODE.S，リスト3をDISP_FLAME.Sとして入力したら，付録ディスクから作成したDISK_4のディレクトリMAGICにコピーしてください。それから5月号で報告したSCRMOD.Sのバグを修正していない方は以下の変更を行ってください。

　ファイル名：SCRMOD.S

　27行subq.w　#1,D1→削除

　28行bne scrmod 2→dbf dl, scrmod 2

そのあと，全ソースファイルをアセンブルしてMAKE.BATを実行すれば，新しいMAGIC.Xが作成されます。

## 若干の注意点

MAGIC.FNCを組み込んでX-BASICを起動すると，表1にあるコマンドが使えるようになります。MAGIC.FNCは，コマンド番号$0008のPOINTを除いたMAGIC.Xのすべてのコマンドを実行する基本コマンド，およびMAGIC.FNCの独自のバッファ操作関係のコマンドで構成されています。

基本コマンドは表1を見れば，だいたいの使い方はわかると思いますが，MAGIC_MODEを使ううえで，若干の注意が必要なので説明しておきます。MAGIC_MODEは今回の改良でXOR, ORモードのみ対応となります。しかし，XORモードの指定はIOCS.Xになって拡張されたICOSコールを利用していますので，ICOS.Xを組み込まないことにはXORモードを利用することはできません。PRESET, NOPについては，MAGIC.Xのバージョンアップで専用ラインルーチンが追加されるはずなので，そのときにサポートしようかと考えています。

## 3D物体を表示するには

ここではX-BASICで3D表示を行う場合の作法を説明していきます。3D物体を表示するための手順は，

1)　ワークスおよび画面初期化
2)　3D物体の形状定義
3)　3Dパラメータの設定
4)　3D→2D変換
5)　表示

となっています。1)，2)について若干の説明が必要なくらいで，難しいことはなにもありません。

まず，1)について説明します。1)を実行するコマンドは，

　magic_init
　magic_screen

の2つです。magic_screenの引数は，表示画面の大きさによって，256×256なら0，512×512なら1を，768×512なら2を与えてください。3D表示をするときは，必ずこれらのコマンドをプログラムの先頭で実行してください。

次に，2)の形状定義は，MAGIC_DATA，MAGIC_PUTBUF，MAGIC_SEEKを組み合わせて行います。まず，物体の形状をint型の配列に設定します。物体の形状データは，もちろんMAGIC.XのコマンドSET 3D DATAとまったく同じ形式です。たとえば，付録ディスクに収録した四角錐を定義するなら以下のようになります。

```
dim int ex(50)
ex={5,   /*頂点の数
      30, 30, -30,
     -30, 30, -30,
     -30, 30,  30,
```

```
30, 30, 30,
 0, −30,  0,
 8,   /＊線分の数
 0,  1,
 1,  2,
 2,  3,
 3,  0,
 0,  4,
 1,  4,
 2,  4,
 3,  4}
```

この例では配列の要素数に適当に大きな値を与えていますが，バッファを有効に使いたいのであれば，要素数をデータの数にあわせるようにしてください。

次に，この形状データをMAGIC. FNCが持っているバッファに定義します。さっきからちょこちょことバッファなんて言葉を使っていますが，いまはなにも考えずに，

obj 1 =magic_buffer(2,ex)

とすると，バッファ2に配列exの内容が格納されるんだということだけ理解してもらえれば十分です。magic_bufferは，配列の格納ポインタをバッファの先頭からのオフセット値で返します。これがobj 1に入りますから，

magic_seek(2,obj 1,0)

とすることで，バッファ2のポインタが格納したデータの先頭に移動します。ここで，

magic_data(2)

を実行すると，バッファ2の現在のポインタから物体の形状データが置かれているものとして，MAGIC. X内部のバッファに3Dデータを格納していきます。

以上が物体の形状を定義するための手順です。ややこしいと感じるかもしれませんが，慣れればどうってことありません。

このあと，3D→2D変換のためのパラメータをMAGIC_PARAで設定してからMAGIC_PERS()，MAGIC_DISP()を実行すれば，MAGIC_COLORで設定した描画色で画面に定義した物体が表示されるはずです。なお3D物体の表示は，グラフィック画面が2ページ以上ある場合に限り，画面切り替えによってちらつきのない表示をします。

リスト1はSIONに出てくるお馴染みのキャラクターを画面に表示して動かすプログラムです。3D物体を表示するプログラムを作る際の参考にしてください。

## バッファについて

3D表示の説明の中で“バッファ”という

表1 MAGIC.FNCリファレンス

```
MAGIC_LINE
 書式   magic_line(x1,y1,x2,y2,palet)
 引数   int
 戻り値 なし
 機能   直線を引く
 バッファ使用量 18バイト
  x1,y1,x2,y2 … -32768~32767
  palet … パレットコード 0~65535
MASIC_CONNECT
 書式   magic_connect(ai)
 引数   int型配列
 戻り値 なし
 機能   連続した直線を引く
 バッファ使用量 頂点の数+6バイト
  備考   ai … { 頂点の数，座標 … 座標 }
MAGIC_SPLINE
 書式   magic_spline(x1,y1,x2,y2,x3,y3,palet)
 引数   int
 戻り値 なし
 機能   3点を結ぶ曲線を描く
 バッファ使用量 20バイト
  x1,y1,x2,y2,x3,y3 … -32768~32767
  palet … パレットコード 0~65535
MAGIC_BOX
 書式   magic_box(x1,y1,x2,y2,palet)
 引数   int
 戻り値 なし
 機能   2点を対角線とするボックスを描く
 バッファ使用量 16バイト
  x1,y1,x2,y2 … -32768~32767
  palet … パレットコード 0~65535
MAGIC_TRIANGLE
 書式   magic_triangle(x1,y1,x2,y2,x3,y3,palet)
 引数   int
 戻り値 なし
 機能   3点を頂点とする三角形を塗り潰す
 バッファ使用量 20バイト
  x1,y1,x2,y2,x3,y3 … -32768~32767
  palet … パレットコード 0~65535
MAGIC_FILL
 書式   magic_fill(x1,y1,x2,y2,palet)
 引数   int
 戻り値 なし
 機能   2点を対角線とするボックスを塗り潰す
 バッファ使用量 16バイト
  x1,y1,x2,y2 … -32768~32767
  palet … パレットコード 0~65535
MAGIC_CIRCLE
 書式   magic_circle(x1,y1,r,palet)
 引数   int
 戻り値 なし
 機能   (x1,y1)を中心として半径 r の円を塗り潰す
 バッファ使用量 14バイト
  x1,y1 … -32768~32767
  r … 半径
  palet … パレットコード 0~65535
MAGIC_WINDOW
 書式   magic_window(x1,y1,x2,y2)
 引数   int
 戻り値 なし
 機能   表示範囲を指定します
 バッファ使用量 12バイト
 備考   座標データは実画面の大きさによって 0~511，0~1023
MAGIC_MODE
 書式   magic_mode(m)
 引数   char
 戻り値 なし
 機能   ラインモードを指定する
 バッファ使用量 6バイト
 m … ラインモード
         0: PRESET
         1: XOR
         2: OR
         3: NOP
  備考  XOR,OR 以外の指定は無視する
MAGIC_WIPE
 書式   magic_wipe()
 引数   なし
 戻り値 なし
 機能   ウィンドウ内をクリアします
 バッファ使用量 4バイト
MAGIC_PARA
 書式   magic_para(para,data)
 引数   char(para),int(data)
 戻り値 なし
 機能   3D-2D変換用のパラメータを設定します
  para … パラメータナンバー
         0 :CX
         1 :CY
         2 :CZ
         3 :DX
         4 :DY
         5 :DZ
         6 :HEAD
         7 :PITCH
         8 :BANK
  data … 設定データ -32768~32767
MAGIC_DATA
 書式   magic_data(buffer)
 引数   char
 戻り値 なし
 機能   物体の3Dデータを設定する
  buffer … バッファ番号 1~5
MAGIC_PERS
 書式   magic_pers()
 引数   なし
 戻り値 なし
 機能   3Dデータを3Dパラメータにしたがって変換し，ワークエリアに格納する
MAGIC_DISP
 書式   magic_disp()
 引数   なし
 戻り値 なし
 機能   ワイヤフレーム表示する
MAGIC_COLOR
 書式   magic_color(palet)
 引数   int
 戻り値 なし
 機能   描画色を設定する
  palet … パレットコード 0~65535
MAGIC_SCREEN
 書式   magic_color(crt)
 引数   char
 戻り値 なし
 機能   画面モードを設定する
  crt … 画面モード
          0 :表示画面256×256
          1 :表示画面512×512
          2 :表示画面768×512
  備考  3D表示する時は必ず実行する
MAGIC_INIT
 書式   magic_init()
 引数   なし
 戻り値 なし
 機能   MAGIC.Xのワークを初期化する
  備考   3D表示する時は必ず実行する
MAGIC_AUTO
 書式   magic_auto(buffer)
 引数   char
 戻り値 int
         >0  実行に使ったバイト数
         -1  バッファにコマンドがない
 機能   バッファのデータを使って自動実行する
  buffer … バッファ番号 1~5
  備考   自動実行するポインタをmagic_seekで指定する
MAGIC_FLUSH
 書式   magic_flush([buffer])
 引数   char
 戻り値 なし
 機能   バッファを初期化する
          省略した場合はすべてのバッファを初期化する
  buffer … バッファ番号 1~5
  備考   必ず実行する
MAGIC_FREE
 書式   magic_free(buffer)
 引数   char
 戻り値 int
 機能   バッファのフリーエリアを返す
  buffer … バッファ番号 1~5
MAGIC_PUTBUF
 書式   magic_putbuf(buffer,ai)
 引数   char(buffer),int型配列(ai)
 戻り値 int
          格納ポインタ
 機能   バッファにデータを格納する
  buffer  … バッファ番号 1~5
  ai … { data }
MAGIC_GETBUF
 書式   magic_gutbuf(buffer)
 引数   char
 戻り値 int
 機能   バッファの内容を返す
  buffer  … バッファ番号 1~5
  ai … { command }
  備考  実行後，ポインタが2つ移動する
MAGIC_SAVE
 書式   magic_save(buffer,filename [,offset] [,size])
 引数   char(buffer),str(filename),int(offset),int(end)
 戻り値 なし
 機能   バッファの内容をセーブする
  buffer   … バッファ番号 1~5
  filename … ファイルネーム
  offset   … バッファの先頭からのオフセット(省略なら0)
  size     … 書込むサイズ(省略ならバッファエンドまで)
MAGIC_SAVEA
 書式   magic_savea(buffer,filename [,offset] [,end])
 引数   char(buffer),str(filename),int(offset),int(end)
 戻り値 なし
 機能   バッファの内容をアスキーセーブする
  buffer   … バッファ番号 1~5
  filename … ファイルネーム
  offset   … バッファの先頭からのオフセット(省略なら0)
  size     … 書込むサイズ(省略ならバッファエンドまで)
MAGIC_LOAD
 書式   magic_load(buffer,filename)
 引数   char(buffer),str(filename)
 戻り値 int 格納ポインタ
 機能   バッファにロードする
  buffer   … バッファ番号 1~5
  filename … ファイルネーム
  備考    バッファの先頭+ポインタからデータが格納される
MAGIC_SEEK
 書式   magic_seek(buffer,offset,mode)
 引数   char(buffer),int(offset),char(mode)
 戻り値 int
          シーク後のポインタ
 機能   バッファのポインタを移動させる
  buffer … バッファ番号 1~5
  offset … オフセット  -65000~65000
  mode   … 0 :バッファの先頭
             1 :現在位置
             2 :バッファエンド
MAGIC_BUFFER
 書式   magic_buffer(buffer)
 引数   char
 戻り値 なし
 機能   読み書きするバッファを指定する
MAGIC_BUFON
 書式   magic_bufon()
 引数   なし
 戻り値 なし
 機能   バッファに保存するようにする
MAGIC_BUFOFF
 書式   magic_bufoff()
 引数   なし
 戻り値 なし
 機能   バッファに保存しないようにする
```

言葉が出てきました。ここでいうバッファとはMAGIC. FNCが内部で持っているコマンドバッファを指しています。MAGICはメモリ上にあるコマンド列を逐次実行していきます。MAGIC. FNCはX-BASICとMAGICのあいだに立ち，BASICから指定されたデータやコマンドをMAGICが理解できるかたちで一時蓄積しておくのです。

バッファは全部で5本あり，各バッファは図1の大きさに固定されています。表1を見てもらうとコマンドによっては，

　バッファ使用量　○○バイト

と，あることに気づくと思いますが，これはこのコマンドが必要とするバッファ使用量を表しています。バッファにコマンドを保存するかどうかは，

　magic_bufon ()
　magic_bufoff ()

で，いつでも切り替えることができますが，バッファに保存するか否かに関係なく，必ずプログラムの先頭で全バッファを初期化するコマンド，

　magic_flush ()

を実行する必要があります。

先ほどの3D表示の説明では触れませんでしたが，3D表示を行う場合であっても，このコマンドを実行する必要があるのです。それはリスト4を見てもらってもわかると思います。

万が一，バッファの初期化を忘れてしまっても，そこはBASICですからエラーメッセージを表示して，プログラムの実行を中止するようになっています。

起動直後はバッファに保存しないモードになっていますので，保存したいのであればmagic_bufon () を実行してください。また，デフォルトではバッファ1にデータが溜められますが，

　magic_buffer(バッファ番号)

で，指定したバッファにデータを保存することができます。

たとえばお絵描きツールを作るとして，バッファにコマンドを残すようにしておけば，ラインの削除や挿入，アンドゥが簡単にできそうです。

また，バッファの内容はディスクに保存することもできます。これには，

　magic_save
　magic_savea

の2つのコマンドがあります。magic_saveはオブジェクト形式，magic_saveaはASCII形式でファイルを作成します。magic_saveaで出力したファイルは，余分なスペースとカンマを取って行頭にdc.wを加えることで，アセンブラのソースリストに埋め込むこともできます。通常はmagic_saveを使うことになるでしょう。

magic_saveでセーブしたファイルをロードするには，

　magic_load

を使います。

このコマンドはバッファのポインタからデータを格納していくものです。読み込み後，ポインタはファイルの大きさだけ移動するので，同じバッファに続けてファイルをロードすると，バッファ内でデータがアペンドされていきます。

バッファ内にあるデータを使って自動実行するコマンドが，

　magic_auto (バッファ番号)

です。このコマンドはバッファ内のポインタから置かれているデータを，MAGIC. Xのコマンド列とみなして，バッファから$000Fが見つかるまで次々とコマンドを実行していくものです。

例として，magic_saveでセーブしたファイルをバッファ1に読み込み，自動実行する手順を紹介しましょう。まず，バッファを初期化するために，

　magic_flush ()

を実行します。こうしてから，

　coml＝magic_load(1，“test. dat”)

とすると，バッファ1の先頭からtest. datの内容が格納されます。なぜ，先頭から格納されるかというと，バッファの初期化によってポインタがバッファの先頭に移動したからです。magic_loadはファイルの格納ポインタをバッファの先頭からのオフセット値で返しますから，comlは0になっているはずです。

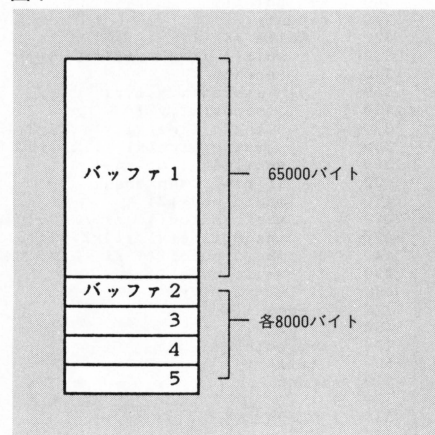
読み込み後ポインタはファイルの大きさだけ移動しています。現在のポインタの位置を知りたいなら，

　print magic_seek(1，0，1)

とすることが確認できます。

さて次に，自動実行するための準備を始めます。まず，ポインタをデータの先頭に

**図1**

バッファ1 — 65000バイト
バッファ2
3
4
5
— 各8000バイト

**リスト1**

```
 10 /* 3D表示サンプル
 20 magic_flush() /* 全バッファ初期化
 30 magic_init()  /* MAGIC.X内部ワーク初期化
 40 magic_screen(1) /* 表示画面512*512
 50 screen 1,2,1,1
 60 int h,obj1
 70 dim int a(250)
 80 a={ 14,
 90     -10,  0,-25,
100     -10,  0, 15,
110      10,  0,-25,
120      10,  0, 15,
130     -10, -5, 15,
140      10, -5, 15,
150     -10,  0, 10,
160      10,  0, 10,
170     -25,  5 ,20,
180      25,  5, 20,
190     -25,  5, 25,
200     -25,  5,  0,
210      25,  5, 25,
220      25,  5,  0,
230      11,
240       0, 1,
250       2, 3,
260       0, 2,
270       1, 3,
280       0, 4,
290       2, 5,
300       4, 5,
310       6, 8,
320       7, 9,
330      11,10,
340      13,12
350 }
360 magic_color(252)  /* 描画色設定
380 for i=0 to 8
390   magic_para(i,0) /* 3Dパラメータクリア
400 next
410 obj1=magic_putbuf(2,a) /* 形状データをバッファに置く
420 magic_seek(2,obj1,0)
430 repeat
440 magic_data(2) /* 物体定義
450 magic_para(2,160) /* Z座標
460 magic_para(6,h) /* head
470 magic_para(7,h) /* pitch
480 magic_para(8,h) /* bank
490 magic_pers() /* 3D→2D変換
500 magic_disp() /* 2D表示
510 h=h+4
520 until inkey$(0)<>""
530 end
```

合わせるために，

magic_seek(1，coml，0)

とします。こうして，

magic_auto(1)

とすることで，バッファの内容を使って自動描画することができます。リスト2に簡単なお絵描きプログラム，リスト3にバッファを使ったプログラムを紹介しておきます。

駆け足でMAGIC. FNCの使い方を説明しました。まだまだ不十分な説明ですが，気力と暇のある方はMAGIC. FNCとサンプルプログラムを入力してみてください。来月以降も引き続きMAGIC. FNCの使い方を説明していきます。

**リスト2**

```
10 /*
20 /* save "sample.bas"
30 /*
40 screen 2,0,1,1          /* 16色4面
50 int x,y,l,r,x1,y1,x2,y2,x3,y3
60 int miny,maxy,com,combak
70 magic_flush()          /* バッファ初期化
80 magic_bufoff()
90 magic_wipe()
100 magic_mode(2)         /* pset モード
110 mouse(4)
120 mouse(1)
130 /*
140 repeat
150   com=asc(inkey$(0)) and &HDF
160   switch chr$(com)
170     case "S":locate 0,0:print"スプライン
              :draw_spline():break
180     case "L":locate 0,0:print"ライン
              :draw_line():break
190     default :locate 0,0:print"(S)pline,(L)ine,(E)nd"
200   endswitch
210 until chr$(com)="E"
220 mouse(0)
230 end
240 /*
250 /* 直線を引く
260 /*
270 func draw_line()
280   magic_bufoff() /* バッファに保存しない
290   magic_mode(1)  /* xor モード
300   set_point()
310   repeat
320     while r<>-1
330       wait() /* ボタンが離されるのを待つ
340       repeat
350         msstat(x,y,l,r)
360         mspos(x2,y2)
370         magic_line(x1,y1,x2,y2,15)
380         magic_line(x1,y1,x2,y2,15)
390       until l=-1 or r=-1
400       if r=-1 then break
410       magic_mode(2)
420       magic_bufon()  /* バッファに保存
430       magic_line(x1,y1,x2,y2,15)
440       magic_bufoff() /* バッファに保存しない
450       magic_mode(1)
460       x1=x2:y1=y2
470     endwhile
480     wait()
490   set_point()
500   until r=-1
510 endfunc
520 /*
530 /* スプラインを引く
540 /*
550 func draw_spline()
560   magic_bufoff() /* バッファに保存しない
570   magic_mode(1)  /* xor モード
580   set_point()
590   repeat
600   while r<>-1
610     wait() /* ボタンが離されるのを待つ
620     repeat
630       msstat(x,y,l,r)
640       mspos(x3,y3)
650       x2=x3:y2=y3
660       magic_spline(x1,y1,x2,y2,x3,y3,15)
670       magic_spline(x1,y1,x2,y2,x3,y3,15)
680     until l=-1 or r=-1
690     wait() /* ボタンが離されるのを待つ
700     repeat
710       msstat(x,y,l,r)
720       mspos(x2,y2)
730       magic_spline(x1,y1,x2,y2,x3,y3,15)
740       magic_spline(x1,y1,x2,y2,x3,y3,15)
750     until l=-1 or r=-1
760     if r=-1 then break
770     magic_mode(2)
780     magic_bufon()  /* バッファに保存
790     magic_spline(x1,y1,x2,y2,x3,y3,15)
800     magic_bufoff() /* バッファに保存しない
810     magic_mode(1)
820     x1=x3:y1=y3
830   endwhile
840   wait()
850   set_point()
860   until r=-1
870 endfunc
880 /*
890 func set_point()
900 wait()
910 repeat
920   mspos(x1,y1)
930   msstat(x,y,l,r)
940 until l=-1 or r=-1
950 return(x1)
960 return(y1)
970 return(l)
980 return(r)
990 endfunc
1000 /*
1010 func wait()
1020 repeat
1030   msstat(x,y,l,r)
1040 until l<>-1 and r<>-1
1050 return(x)
1060 return(y)
1070 return(l)
1080 return(r)
1090 endfunc
1100 /* ファイルセーブ
1110 func ds()
1120   magic_save(1,"sample.dat")
1130 endfunc
1140 /* ファイルロード
1150 func dl()
1155 int a,point
1157   wipe()
1160   magic_flush(2)
1170   point=magic_load(2,"sample.dat")
1180   magic_seek(2,point,0)
1190   repeat
1200     a=magic_auto(2)
1210   until a=-1
1220 endfunc
```

**リスト3**

```
10 /*
20 /* save "sample2.bas"
30 /*
40 screen 1,2,1,1
50 magic_flush()
60 magic_bufon()
70 for i=1 to 80
80   x1=int(rnd()*512)
90   x2=int(rnd()*512)
100  x3=int(rnd()*512)
110  y1=int(rnd()*512)
120  y2=int(rnd()*512)
130  y3=int(rnd()*512)
140  col=int(rnd()*256)
150  magic_buffer(1)
160  magic_line(x1,y1,x2,y2,col)
170  magic_buffer(2)
180  magic_box(x1,y1,x2,y2,col)
190  magic_buffer(3)
200  magic_spline(x1,y1,x2,y2,x3,y3,col)
210  magic_buffer(4)
220  magic_triangle(x1,y1,x2,y2,x3,y3,col)
230  magic_buffer(5)
240  magic_circle(x2,y2,int(rnd()*100),int(rnd()*256))
250 next
260 disp():goto 260
270 end
280 /*
290 func disp()
300 int a
310 char buffer
320 wipe()
330 print"どのバッファを選択しますか(1～5)"
340 repeat
350   buffer=asc(inkey$(0))-&H30
360 until buffer>0 and buffer<6
370 magic_seek(buffer,0,0)
380 repeat
390   a=magic_auto(buffer)
400 until a=-1
410 endfunc
```

## リスト4

```
  1: *
  2: * magic_fnc.s    1991/5
  3: *
  4: *                       written by Hiroaki Kageyama
  5: *
  6: * MAGICをX-BASICから使う
  7: *
  8:         .include        iocscall.mac
  9:         .include        doscall.mac
 10:         .include        magic.mac
 11:         .include        magic.h
 12:         .include        fdef.h
 13:
 14: combuf          equ     65000
 15: bufsize         equ     8000
 16: track           equ     5
 17:
 18:         .text
 19:         .even
 20:
 21:         dc.l    x_init
 22:         dc.l    x_run
 23:         dc.l    x_end
 24:         dc.l    x_sys
 25:         dc.l    x_brk
 26:         dc.l    x_ctrl_d
 27:         dc.l    x_res1
 28:         dc.l    x_res2
 29:         dc.l    ptr_token
 30:         dc.l    ptr_param
 31:         dc.l    ptr_exec
 32:         dc.l    0,0,0,0,0
 33:
 34: ****************************************
 35: * ライン
 36: ****************************************
 37: x_line:
 38:         tst.w   flush_flg
 39:         bmi     flush_err       * バッファが初期化されていない
 40:         bsr     g_chk
 41:         bmi     g_err           * グラフィック画面が使えない
 42:         lea.l   parameter,a0
 43:         tst.w   buf_flg
 44:         bmi     x_line2         * バッファに保存しない
 45:         move.w  buf_no,d0       * バッファ番号
 46:         move.w  #9,d1           * 必要なワード数
 47:         bsr     chkbuf
 48:         bcs     no_buf_err      * バッファが足りない
 49:         bsr     get_bufadrs
 50:         movea.l a0,a6           * パラメータ格納アドレスが置かれたアドレス
 51:         movea.l (a0),a0
 52: x_line2:
 53:         movea.l a0,a1
 54:
 55:         move.l  52(sp),d1       * 描画色
 56:         bsr     chk_color
 57:         tst.l   d0
 58:         bne     x_line_end      * 無効な描画色だ
 59:         move.w  #COLOR,(a1)+    * カラー
 60:         move.w  d1,(a1)+        * 描画色
 61:
 62:         bsr     get_fourparam   * 座標は4つ
 63:         tst.l   d0
 64:         bne     x_line_end      * 座標がおかしい
 65:         move.w  #LINE,(a1)+     * スプライン
 66:         move.w  #2,(a1)+
 67:         movem.w d1-d4,(a1)      * 座標
 68:         addq.l  #8,a1
 69:         move.w  #DONE,(a1)+     * 終了
 70:         move.l  a0,(a6)
 71:         tst.w   buf_flg
 72:         bmi     x_line3
 73:         move.l  a1,(a6)
 74: x_line3:
 75:         MAGIC   __AUTO
 76:         clr.l   d0
 77: x_line_end:
 78:         rts
 79:
 80: ****************************************
 81: * コネクトライン
 82: ****************************************
 83: x_connect:
 84:         tst.w   flush_flg
 85:         bmi     flush_err       * バッファが初期化されていない
 86:         bsr     g_chk
 87:         bmi     g_err           * グラフィック画面が使えない
 88:         movea.l 12(sp),a1
 89:         lea.l   10(a1),a2       * データの先頭アドレス
 90:         move.l  (a2),d1
 91:         add.w   d1,d1           * 頂点の数
 92:         lea.l   parameter,a0
 93:         tst.w   buf_flg
 94:         bmi     x_connect2      * バッファに保存しない
 95:         move.w  buf_no,d0       * バッファ番号
 96:         addq.w  #3,d1           * 必要なワード数
 97:         bsr     chkbuf
 98:         bcs     no_buf_err      * バッファが足りない
 99:         bsr     get_bufadrs
100:         movea.l a0,a6           * パラメータ格納アドレスが置かれたアドレス
101:         movea.l (a0),a0
102: x_connect2:
103:         movea.l a0,a1
104:
105:         move.w  #LINE,(a1)+     * ライン
106: x_connect3:
107:         move.l  (a2)+,d1        * 1個取得
108:         move.w  d1,(a1)+        * パラメータをワードに加工
109:         dbf     d0,x_connect3   * まだパラメータがある
110:         move.w  #DONE,(a1)+
111:         move.l  a0,(a6)
112:         tst.w   buf_flg
113:         bmi     x_connect4
114:         move.l  a1,(a6)
115: x_connect4:
116:         MAGIC   __AUTO
117:         clr.l   d0              * 正常終了
118:         rts
119: ****************************************
120: * スプライン
121: ****************************************
122: x_spline:
123:         tst.w   flush_flg
124:         bmi     flush_err       * バッファが初期化されていない
125:         bsr     g_chk
126:         bmi     g_err           * グラフィック画面が使えない
127:         lea.l   parameter,a0
128:         tst.w   buf_flg
129:         bmi     x_spline2       * バッファに保存しない
130:         move.w  buf_no,d0       * バッファ番号
131:         move.w  #10,d1          * 必要なワード数
132:         bsr     chkbuf
133:         bcs     no_buf_err      * バッファが足りない
134:         bsr     get_bufadrs
135:         movea.l a0,a6           * パラメータ格納アドレスが置かれたアドレス
136:         movea.l (a0),a0
137: x_spline2:
138:         movea.l a0,a1
139:
140:         move.l  72(sp),d1       * 描画色
141:         bsr     chk_color
142:         tst.l   d0
143:         bne     x_spline_end    * 無効な描画色だ
144:         move.w  #COLOR,(a1)+    * カラー
145:         move.w  d1,(a1)+        * 描画色
146:
147:         bsr     get_sixparam    * 座標は6つ
148:         tst.l   d0
149:         bne     x_spline_end    * 座標がおかしい
150:         move.w  #SPLINE,(a1)+   * スプライン
151:         movem.w d1-d6,(a1)      * 座標
152:         add.w   #12,a1
153:         move.w  #DONE,(a1)+     * 終了
154:         move.l  a0,(a6)
155:         tst.w   buf_flg
156:         bmi     x_spline3
157:         move.l  a1,(a6)
158: x_spline3:
159:         MAGIC   __AUTO
160:         clr.l   d0
161: x_spline_end:
162:         rts
163: ****************************************
164: * ボックス
165: ****************************************
166: x_box:
167:         tst.w   flush_flg
168:         bmi     flush_err       * バッファが初期化されていない
169:         bsr     g_chk
170:         bmi     g_err           * グラフィック画面が使えない
171:         lea.l   parameter,a0
172:         tst.w   buf_flg
173:         bmi     x_box2          * バッファに保存しない
174:         move.w  buf_no,d0       * バッファ番号
175:         move.w  #8,d1           * 必要なワード数
176:         bsr     chkbuf
177:         bcs     no_buf_err      * バッファが足りない
178:         bsr     get_bufadrs
179:         movea.l a0,a6           * パラメータ格納アドレスが置かれたアドレス
180:         movea.l (a0),a0
181: x_box2:
182:         movea.l a0,a1
183:
184:         move.l  52(sp),d1       * 描画色
185:         bsr     chk_color
186:         tst.l   d0
187:         bne     x_box_end
188:         move.w  #COLOR,(a1)+    * カラー
189:         move.w  d1,(a1)+        * 描画色
190:
191:         bsr     get_fourparam
192:         tst.l   d0
193:         bne     x_box_end
194:         move.w  #BOX,(a1)+      * スプライン
195:         movem.w d1-d4,(a1)      * 座標
196:         addq.l  #8,a1
197:         move.w  #DONE,(a1)+     * 終了
198:         move.l  a0,(a6)
199:         tst.w   buf_flg
200:         bmi     x_box3
201:         move.l  a1,(a6)
202: x_box3:
203:         MAGIC   __AUTO
204:         clr.l   d0
205: x_box_end:
206:         rts
207: ****************************************
208: * トライアングル
209: ****************************************
210: x_triangle:
211:         tst.w   flush_flg
212:         bmi     flush_err       * バッファが初期化されていない
213:         bsr     g_chk
214:         bmi     g_err           * グラフィック画面が使えない
215:         lea.l   parameter,a0
216:         tst.w   buf_flg
217:         bmi     x_triangle2     * バッファに保存する
218:         move.w  buf_no,d0       * バッファ番号
219:         move.w  #10,d1          * 必要なワード数
220:         bsr     chkbuf
221:         bcs     no_buf_err      * バッファが足りない
222:         bsr     get_bufadrs
223:         movea.l a0,a6           * パラメータ格納アドレスが置かれたアドレス
224:         movea.l (a0),a0
225: x_triangle2:
226:         movea.l a0,a1
227:
228:         move.l  72(sp),d1       * 描画色
229:         bsr     chk_color
230:         tst.l   d0
231:         bne     x_triangle_end  * 無効な描画色だ
232:         move.w  #COLOR,(a1)+    * カラー
233:         move.w  d1,(a1)+        * 描画色
234:
235:         bsr     get_sixparam    * 座標は6つ
236:         tst.l   d0
237:         bne     x_triangle_end  * 座標がおかしい
238:         move.w  #TRIANGLE,(a1)+ * スプライン
239:         movem.w d1-d6,(a1)      * 座標
240:         add.w   #12,a1
241:         move.w  #DONE,(a1)+     * 終了
242:         move.l  a0,(a6)
243:         tst.w   buf_flg
244:         bmi     x_triangle3
245:         move.l  a1,(a6)
246: x_triangle3:
247:         MAGIC   __AUTO
248:         clr.l   d0
249: x_triangle_end:
250:         rts
251: ****************************************
252: * ボックスフル
```

```
253: ****************************************
254: x_boxfull:
255:         tst.w   flush_flg
256:         bmi     flush_err          * バッファが初期化されていない
257:         bsr     g_chk
258:         bmi     g_err              * グラフィック画面が使えない
259:         lea.l   parameter,a0
260:         tst.w   buf_flg
261:         bmi     x_boxfull2         * バッファに保存しない
262:         move.w  buf_no,d0          * バッファ番号
263:         move.w  #8,d1              * 必要なワード数
264:         bsr     chkbuf
265:         bcs     no_buf_err         * バッファが足りない
266:         bsr     get_bufadrs
267:         movea.l a0,a6              * パラメータ格納アドレスが置かれたアドレス
268:         movea.l (a0),a0
269: x_boxfull2:
270:         movea.l a0,a1
271:
272:         move.l  52(sp),d1          * 描画色
273:         bsr     chk_color
274:         tst.l   d0
275:         bne     x_boxfull_end
276:         move.w  #COLOR,(a1)+       * カラー
277:         move.w  d1,(a1)+           * 描画色
278:
279:         bsr     get_fourparam
280:         tst.l   d0
281:         bne     x_boxfull_end
282:         move.w  #FILL,(a1)+        * スプライン
283:         movem.w d1-d4,(a1)         * 座標
284:         addq.l  #8,a1
285:         move.w  #DONE,(a1)+        * 終了
286:         move.l  a0,(a6)
287:         tst.w   buf_flg
288:         bmi     x_boxfull3
289:         move.l  a1,(a6)
290: x_boxfull3:
291:         MAGIC   __AUTO
292:         clr.l   d0
293: x_boxfull_end:
294:         rts
295: ****************************************
296: * サークルフル
297: ****************************************
298: x_circle:
299:         tst.w   flush_flg
300:         bmi     flush_err          * バッファが初期化されていない
301:         bsr     g_chk
302:         bmi     g_err              * グラフィック画面が使えない
303:         lea.l   parameter,a0
304:         tst.w   buf_flg
305:         bmi     x_circle2          * バッファに保存しない
306:         move.w  buf_no,d0          * バッファ番号
307:         move.w  #7,d1              * 必要なワード数
308:         bsr     chkbuf
309:         bcs     no_buf_err         * バッファが足りない
310:         bsr     get_bufadrs
311:         movea.l a0,a6              * パラメータ格納アドレスが置かれたアドレス
312:         movea.l (a0),a0
313: x_circle2:
314:         movea.l a0,a1
315:
316:         move.l  42(sp),d1          * 描画色
317:         bsr     chk_color
318:         tst.l   d0
319:         bne     x_circle_end
320:         move.w  #COLOR,(a1)+       * カラー
321:         move.w  d1,(a1)+           * 描画色
322:
323:         bsr     get_threeparam
324:         tst.l   d0
325:         bne     x_circle_end
326:         move.w  #CIRCLE,(a1)+      * スプライン
327:         movem.w d1-d3,(a1)         * 座標
328:         addq.l  #6,a1
329:         move.w  #DONE,(a1)+        * 終了
330:         move.l  a0,(a6)
331:         tst.w   buf_flg
332:         bmi     x_circle3
333:         move.l  a1,(a6)
334: x_circle3:
335:         MAGIC   __AUTO
336:         clr.l   d0
337: x_circle_end:
338:         rts
339: ****************************************
340: * ウィンドウ
341: ****************************************
342: x_window:
343:         tst.w   flush_flg
344:         bmi     flush_err          * バッファが初期化されていない
345:         bsr     g_chk
346:         bmi     g_err              * グラフィック画面が使えない
347:         lea.l   parameter,a0
348:         tst.w   buf_flg
349:         bmi     x_window2          * バッファに保存しない
350:         move.w  buf_no,d0          * バッファ番号
351:         move.w  #6,d1              * 必要なワード数
352:         bsr     chkbuf
353:         bcs     no_buf_err         * バッファが足りない
354:         bsr     get_bufadrs
355:         movea.l a0,a6              * パラメータ格納アドレスが置かれたアドレス
356:         movea.l (a0),a0
357: x_window2:
358:         movea.l a0,a1
359:
360:         move.w  #1023,d5
361:         move.w  #-1,d1
362:         IOCS    _CRTMOD
363:         cmp.w   #4,d0
364:         bcs     x_window3          * 実画面1024×1024
365:         cmp.w   #16,d0
366:         beq     x_window3
367:         move.w  #511,d5            * 実画面512×512
368: x_window3:
369:         move.w  14(sp),d1          * minx
370:         move.w  24(sp),d2          * miny
371:         move.w  34(sp),d3          * maxx
372:         move.w  44(sp),d4          * maxy
373:
374:         cmp.w   #1024,d1
375:         bcc     zahyou_err
376:         cmp.w   #1024,d2
377:         bcc     zahyou_err
378:         cmp.w   #1024,d3
379:         bcc     zahyou_err
380:         cmp.w   #1024,d4
381:         bcc     zahyou_err
382:
383:         cmp.w   d5,d1
384:         bhi     zahyou_err2
385:         cmp.w   d5,d2
386:         bhi     zahyou_err2
387:         cmp.w   d5,d3
388:         bhi     zahyou_err2
389:         cmp.w   d5,d4
390:         bhi     zahyou_err2
391:
392:         move.w  #WINDOW,(a1)+
393:         movem.w d1-d4,(a1)
394:         addq.l  #8,a1
395:         move.w  #DONE,(a1)+
396:         move.l  a0,(a6)
397:         tst.w   buf_flg
398:         bmi     x_window4
399:         move.l  a1,(a6)
400: x_window4:
401:         MAGIC   __AUTO
402:         clr.l   d0
403:         rts
404: ****************************************
405: * 描画モード
406: ****************************************
407: x_mode:
408:         tst.w   flush_flg
409:         bmi     flush_err          * バッファが初期化されていない
410:         lea.l   parameter,a0
411:         tst.w   buf_flg
412:         bmi     x_mode2            * バッファに保存しない
413:         move.w  buf_no,d0          * バッファ番号
414:         move.w  #3,d1              * 必要なワード数
415:         bsr     chkbuf
416:         bcs     no_buf_err         * バッファが足りない
417:         bsr     get_bufadrs
418:         movea.l a0,a6              * パラメータ格納アドレスが置かれたアドレス
419:         movea.l (a0),a0
420: x_mode2:
421:         movea.l a0,a1
422:
423:         move.w  14(sp),d1          * 描画モード
424:         cmpi.w  #4,d1
425:         bcc     mode_err
426:
427:         move.w  #MODE,(a1)+
428:         move.w  d1,(a1)+
429:         move.w  #DONE,(a1)+
430:         move.l  a0,(a6)
431:         tst.w   buf_flg
432:         bmi     x_mode3
433:         move.l  a1,(a6)
434: x_mode3:
435:         MAGIC   __AUTO
436:         clr.l   d0
437:         rts
438: ****************************************
439: * グラフィック画面クリア
440: ****************************************
441: x_wipe:
442:         tst.w   flush_flg
443:         bmi     flush_err          * バッファが初期化されていない
444:         lea.l   parameter,a0
445:         tst.w   buf_flg
446:         bmi     x_wipe2            * バッファに保存しない
447:         move.w  buf_no,d0          * バッファ番号
448:         move.w  #2,d1              * 必要なワード数
449:         bsr     chkbuf
450:         bcs     no_buf_err         * バッファが足りない
451:         bsr     get_bufadrs
452:         movea.l a0,a6              * パラメータ格納アドレスが置かれたアドレス
453:         movea.l (a0),a0
454: x_wipe2:
455:         movea.l a0,a1
456:
457:         move.w  #CLS,(a1)+
458:         move.w  #DONE,(a1)+
459:         move.l  a0,(a6)
460:         tst.w   buf_flg
461:         bmi     x_wipe3
462:         move.l  a1,(a6)
463: x_wipe3:
464:         MAGIC   __AUTO
465:         clr.l   d0
466:         rts
467: ****************************************
468: * 3Dパラメータ設定
469: ****************************************
470: x_para:
471:         tst.w   init_flg
472:         bmi     init_err
473:         lea.l   parameter,a0
474:         move.w  14(sp),d1
475:         cmpi.w  #9,d1
476:         bcc     para_err
477:         move.l  22(sp),d2
478:         cmpi.l  #$10000,d2
479:         bcc     mukou_err
480:         move.w  d1,(a0)
481:         move.w  d2,2(a0)
482:         MAGIC   __3D_PARM
483:         clr.l   d0
484:         rts
485: ****************************************
486: * 3Dデータ設定
487: ****************************************
488: x_data:
489:         tst.w   init_flg
490:         bmi     init_err           * バッファが初期化されていない
491:         move.w  14(sp),d0
492:         bsr     chkbuf_no
493:         bcs     para_err           * バッファ番号が不適
494:         bsr     get_paramadrs
495:         subq.w  #1,d0
496:         add.w   d0,d0
497:         lea.l   buf_point,a3
498:         clr.l   d3
499:         move.w  (a3,d0.w),d3
500:         adda.l  d3,a0              * 格納アドレス
501:         MAGIC   __3D_DATA
502:         clr.l   d0                 * 正常終了
503:         rts
504: ****************************************
505: * 3D→2D変換
506: ****************************************
507: x_pers:
508:         tst.w   init_flg
509:         bmi     init_err
510:         MAGIC   __3D_TRANS
```

▶MAGICがX68000に移植！　これはアセンブラが理解できるようになる絶好の機会でしょう。……あ，しまった私は受験生だあ。　光石　和弘(18)神奈川県

```
511:          clr.l   d0
512:          rts
513: ****************************************
514: * 2D表示
515: ****************************************
516: x_disp_flame:
517:          tst.w   init_flg
518:          bmi     init_err
519:          bsr     g_chk
520:          bmi     g_err               * グラフィック画面が使えない
521:          MAGIC   __DISPLAY
522:          clr.l   d0
523:          rts
524: ****************************************
525: * 描画色設定
526: ****************************************
527: x_color:
528:          tst.w   flush_flg
529:          bmi     flush_err           * バッファが初期化されていない
530:          lea.l   parameter,a0
531:          tst.w   buf_flg
532:          bmi     x_color2            * バッファに保存しない
533:          move.w  buf_no,d0           * バッファ番号
534:          move.w  #3,d1               * 必要なワード数
535:          bsr     chkbuf
536:          bcs     no_buf_err          * バッファが足りない
537:          bsr     get_bufadrs
538:          movea.l a0,a6               * パラメータ格納アドレスが置かれたアドレス
539:          movea.l (a0),a0
540: x_color2:
541:          movea.l a0,a1
542:
543:          move.l  12(sp),d1           * 描画色
544:          bsr     chk_color
545:          tst.l   d0
546:          bne     x_color_end
547:
548:          move.w  #COLOR,(a1)+
549:          move.w  d1,(a1)+
550:          move.w  #DONE,(a1)+
551:          move.l  a0,(a6)
552:          tst.w   buf_flg
553:          bmi     x_color3
554:          move.l  a1,(a6)
555: x_color3:
556:          MAGIC   __AUTO
557:          clr.l   d0
558: x_color_end:
559:          rts
560: ****************************************
561: * 画面モード設定
562: ****************************************
563: x_screen:
564:          lea.l   parameter,a0
565:          move.w  14(sp),(a0)
566:          MAGIC   __CRT
567:          clr.l   d0
568:          rts
569: ****************************************
570: * マジック初期化
571: ****************************************
572: x_minit:
573:          MAGIC   __INIT
574:          clr.w   init_flg
575:          clr.l   d0
576:          rts
577: ****************************************
578: * 自動実行
579: ****************************************
580: x_auto:
581:          tst.w   flush_flg
582:          bmi     flush_err           * バッファが初期化されていない
583:          bsr     g_chk
584:          bmi     g_err               * グラフィック画面が使えない
585:          move.w  14(sp),d0
586:          bsr     chkbuf_no
587:          bcs     para_err            * バッファ番号が不適
588:          bsr     get_paramadrs       * パラメータ格納先頭アドレス
589:
590:          move.l  #-1,int_dat         * 戻り値 -1 にしておく
591:          lea.l   buf_use,a1
592:          lea.l   buf_point,a2
593:          subq.w  #1,d0               * バッファ番号－1
594:          add.w   d0,d0
595:          move.w  (a1,d0.w),d2
596:          clr.l   d3
597:          move.w  (a2,d0.w),d3        * ポインタ
598:          cmp.w   d3,d2
599:          bls     x_auto_end          * buf_use≦buf_point
600:          adda.l  d3,a0               * +ポインタ
601:          movea.l a0,a1               * a0 保存
602:
603:          move.w  (a0),d3
604:          cmp.w   #$13,d3             * あってはいけない
605:          beq     not_com_err
606:          cmp.w   #$13+1,d3           * 13H以上は未定義
607:          bcc     not_com_err
608:
609:          MAGIC   __AUTO
610:          suba.l  a1,a0
611:          move.l  a0,d1               * ポインタの移動量
612:          add.w   d1,(a2,d0.w)        * ワークに格納
613:          move.l  d1,int_dat
614: x_auto_end:
615:          lea.l   ret_dat,a0
616:          clr.l   d0
617:          rts
618: ****************************************
619: * バッファフラッシュする
620: *
621: * 引数：バッファ番号
622: *        省略 … すべてのバッファを対象にする
623: *
624: * 戻り値：なし
625: ****************************************
626: x_flush:
627:          clr.w   flush_flg           * バッファを初期化したことを示すフラグ
628:          tst.w   6(sp)
629:          bmi     x_flush2            * 引数が省略された
630:          move.w  14(sp),d0           * バッファ番号
631:          bsr     chkbuf_no
632:          bcs     para_err            * バッファ番号が不適
633:          bsr     get_paramadrs       * パラメータ格納先頭アドレスを a0 に求める
634:          lea.l   buf_use,a1
635:          lea.l   buf_point,a2
636:          lea.l   param_adrs,a3
637:          subq.w  #1,d0
638:          add.w   d0,d0
639:          clr.w   (a1,d0.w)
640:          clr.w   (a2,d0.w)
641:          add.w   d0,d0
642:          move.l  a0,(a3,d0.w)        * バッファ先頭アドレス
643:          move.w  #DONE,(a0)          * 先頭に DONE を書いておく
644:          clr.l   d0
645:          rts
646: ****************************************
647: * 引数が省略された場合
648: ****************************************
649: x_flush2:
650:          move.w  #1,d0
651:          clr.w   d1
652:          clr.w   d2
653:          clr.l   d3
654:          lea.l   buf_use,a1
655:          lea.l   buf_point,a2
656:          lea.l   param_adrs,a3
657:          lea.l   param_format,a4
658:          lea.l   param_work,a5
659: x_flush3:
660:          clr.w   (a1)+
661:          clr.w   (a2)+
662:          adda.l  d3,a5
663:          move.l  a5,(a3,d2.w)
664:          move.w  #DONE,(a5)
665:          move.w  (a4,d1.w),d3
666:          addq.w  #1,d0
667:          addq.w  #2,d1
668:          addq.w  #4,d2
669:          cmpi.w  #track,d0
670:          bls     x_flush3
671:          clr.l   d0
672:          rts
673: ****************************************
674: * バッファにデータを格納する
675: ****************************************
676: x_putbuf:
677:          tst.w   flush_flg
678:          bmi     flush_err           * バッファが初期化されていない
679:          move.w  14(sp),d0
680:          bsr     chkbuf_no
681:          bcs     para_err            * 無効なバッファ番号だ
682:          bsr     get_paramadrs       * パラメータ格納先頭アドレス
683:          move.w  d0,-(sp)            * バッファ番号退避
684:          subq.w  #1,d0
685:          add.w   d0,d0
686:          lea.l   buf_point,a3
687:          clr.l   d3
688:          move.w  (a3,d0.w),d3        * ポインタ
689:          move.w  (sp)+,d0            * バッファ番号復帰
690:          move.l  d3,int_dat          * 戻り値
691:          adda.l  d3,a0
692:          movea.l 16+6(sp),a2
693:          move.w  8(a2),d1            * 要素数
694:          move.w  d1,d7               * 要素数退避
695:          bsr     chkbuf
696:          bcs     no_buf_err          * バッファがたりない
697:          lea.l   10(a2),a2           * データの先頭アドレス
698: x_putbuf2:
699:          move.l  (a2)+,d2
700:          move.w  d2,(a0)+
701:          dbf     d7,x_putbuf2
702:          lea.l   ret_dat,a0
703:          clr.l   d0
704:          rts
705: ****************************************
706: * バッファからデータを得る
707: ****************************************
708: x_getbuf:
709:          tst.w   flush_flg
710:          bmi     flush_err
711:          move.w  14(sp),d0
712:          bsr     chkbuf_no
713:          bcs     para_err
714:          bsr     get_paramadrs       * パラメータ格納先頭アドレスを求める
715:          lea.l   buf_point,a1
716:          lea.l   buf_use,a2
717:          subq.w  #1,d0
718:          add.w   d0,d0
719:          clr.l   d1
720:          move.w  (a1,d0.w),d1
721:          adda.l  d1,a0
722:          cmp.w   (a2,d0.w),d1        * buf_point≧buf_use
723:          bcc     bufend_err
724:          addq.w  #2,d1               * ポインタを2つ進める
725:          move.w  d1,(a1,d0.w)
726: x_getbuf2:
727:          clr.l   int_dat
728:          move.w  (a0),int_dat+2
729:          lea.l   ret_dat,a0
730:          clr.l   d0
731:          rts
732: ****************************************
733: * 格納バッファを指定する
734: ****************************************
735: x_buffer:
736:          move.l  12(sp),d0
737:          bsr     chkbuf_no
738:          bcs     para_err
739:          move.w  d0,buf_no           * バッファ番号保存
740:          clr.l   d0
741:          rts
742: ****************************************
743: * バッファに格納する
744: ****************************************
745: x_bufon:
746:          clr.w   buf_flg
747:          clr.l   d0
748:          rts
749: ****************************************
750: * バッファに格納しない
751: ****************************************
752: x_bufoff:
753:          move.w  #-1,buf_flg
754:          clr.l   d0
755:          rts
756: ****************************************
757: * バッファロード
758: ****************************************
759: x_load:
760:          tst.w   flush_flg
761:          bmi     flush_err
762:          move.w  14(sp),d0
763:          bsr     chkbuf_no
764:          bcs     para_err            * バッファ番号が無効
765:          bsr     get_paramadrs
766:          lea.l   buf_point,a2
767:          subq.w  #1,d0               * バッファ番号－1
768:          add.w   d0,d0
769:          adda.w  d0,a2
```

▶6月号の「大人のためのX68000」第8回の3行目の1文は多くの国民の声を代弁していると思う。

倉田　泰幸(21)茨城県

```
769:            adda.w  d0,a2
770:            clr.l   d0
771:            move.w  (a2),d0
772:            adda.l  d0,a0
773:
774:            movea.l 22(sp),a3       * ファイルネームのポインタ
775:            clr.w   -(sp)           * 読み込みモード
776:            move.l  a3,-(sp)
777:            DOS     _OPEN           * ファイルオープン
778:            addq.l  #6,sp
779:            tst.l   d0
780:            bmi     open_err        * ファイルが作成できない
781:            move.w  d0,handle       * ファイルハンドルを退避
782:
783:            move.l  #combuf,-(sp)
784:            tst.w   d1
785:            beq     x_load2
786:            move.l  #bufsize,(sp)
787: x_load2:
788:            move.l  a0,-(sp)
789:            move.w  d0,-(sp)        * ファイルハンドル
790:            DOS     _READ
791:            lea.l   10(sp),sp
792:            tst.l   d0
793:            bmi     read_err        * 読み込みエラー
794:            adda.l  d0,a0
795:            clr.l   int_dat
796:            move.w  (a2),int_dat+2
797:            move.w  d0,d1
798:            asr.w   #1,d1           * 必要なワード数
799:            move.w  14(sp),d0       * バッファ番号
800:            bsr     chkbuf
801:            bcs     no_buf_err      * バッファが足りない
802:            move.w  handle,-(sp)
803:            DOS     _CLOSE
804:            addq.l  #2,sp
805:            lea.l   ret_dat,a0
806:            clr.l   d0
807:            rts
808: ****************************************
809: * バッファセーブ
810: ****************************************
811: x_save:
812:            tst.w   flush_flg
813:            bmi     flush_err
814:            move.w  14(sp),d0
815:            bsr     chkbuf_no
816:            bcs     para_err        * バッファ番号が無効
817:            bsr     get_paramadrs   * パラメータ格納先頭アドレスを求める
818:            lea.l   buf_use,a2
819:            subq.w  #1,d0
820:            add.w   d0,d0
821:            clr.l   d1
822:            move.w  (a2,d0.w),d1    * バッファサイズ
823:            movea.l 22(sp),a3       * ファイルネームのポインタ
824:            move.w  #$20,-(sp)      * アーカイブファイル
825:            move.l  a3,-(sp)
826:            DOS     _CREATE         * ファイル作成
827:            addq.l  #6,sp
828:            tst.l   d0
829:            bmi     create_err      * ファイルが作成できない
830:            move.w  d0,handle       * ファイルハンドルを退避
831:            clr.l   d2              * オフセット0
832:            tst.w   26(sp)          * 先頭からのオフセット
833:            bmi     x_save2         * 省略された
834:            move.l  32(sp),d2       * オフセット
835:            adda.l  d2,a0           * 先頭アドレス+オフセット
836: x_save2:
837:            sub.l   d2,d1           * デフォルトの書込みサイズ
838:            tst.w   36(sp)          * 書き込むサイズ
839:            bmi     x_save3         * 省略された
840:            move.l  42(sp),d1       * 書込むサイズが指定された
841: x_save3:
842:            move.l  d1,-(sp)        * 書き込むサイズ
843:            move.l  a0,-(sp)        * 先頭アドレス
844:            move.w  d0,-(sp)        * ファイルハンドル
845:            DOS     _WRITE
846:            lea.l   10(sp),sp
847:            cmp.l   d0,d1
848:            bne     write_err       * 総てのデータを書き込めなかった
849:            tst.l   d0
850:            bmi     write_err       * 書き込みエラー
851:            move.w  handle,-(sp)
852:            DOS     _CLOSE
853:            addq.l  #2,sp
854:            clr.l   d0
855:            rts
856: ****************************************
857: * バッファセーブ
858: ****************************************
859: x_save@:
860:            tst.w   flush_flg
861:            bmi     flush_err
862:            move.w  14(sp),d0
863:            bsr     chkbuf_no
864:            bcs     para_err        * バッファ番号が無効
865:            bsr     get_paramadrs   * パラメータ格納先頭アドレスを求める
866:            lea.l   buf_use,a2
867:            subq.w  #1,d0
868:            add.w   d0,d0
869:            move.w  (a2,d0.w),d2    * バッファサイズ
870:            clr.l   d1
871:            tst.w   26(sp)          * 先頭からのオフセット
872:            bmi     x_save@2        * 省略された
873:            move.l  32(sp),d1       * オフセット
874:            adda.l  d1,a0           * 先頭アドレス+オフセット
875: x_save@2:
876:            sub.l   d1,d2           * デフォルトの書込みサイズ
877:            tst.w   36(sp)          * 書き込むサイズ
878:            bmi     x_save@3        * 省略された
879:            move.l  42(sp),d2       * 書込むサイズが指定された
880: x_save@3:
881:            asr.w   #1,d2
882:            movea.l 22(sp),a3       * ファイルネームのポインタ
883:            move.w  #$20,-(sp)      * アーカイブファイル
884:            move.l  a3,-(sp)
885:            DOS     _CREATE         * ファイル作成
886:            addq.l  #6,sp
887:            tst.l   d0
888:            bmi     create_err      * ファイルが作成できない
889:            move.w  d0,handle       * ファイルハンドルを退避
890:            tst.w   d2
891:            beq     convstr10
892:            subq.w  #1,d2           * バッファサイズ-2
893:            lea.l   format,a2
894:            clr.w   d6
895: x_save@4:
896:            lea.l   ascii,a1
897:            clr.w   d7
898: convstr:
899:            move.w  (a0)+,d1
900:            bne     convstr1        * 0だ
901:            move.l  #'    ',(a1)+
902:            move.w  #' +',(a1)+
903:            move.b  #'0',(a1)
904:            bra     convstr8
905: convstr1:
906:            bpl     convstr2
907:            smi     d7              * フラグ=1
908:            neg.w   d1
909: convstr2:
910:            clr.w   d3
911:            move.l  #10000,d4
912:            move.w  #10,d5
913:            move.w  #5,d0
914: convstr3:
915:            cmp.w   d4,d1
916:            bcc     convstr4
917:            divu.w  d5,d4
918:            addq.w  #1,d3
919:            dbf     d0,convstr3
920: convstr4:
921:            move.b  #' ',(a1)+      * スペース
922:            dbf     d3,convstr4
923:
924:            move.b  #'+',d3
925:            tst.w   d7
926:            beq     convstr6
927:            move.b  #'-',d3
928: convstr6:
929:            move.b  d3,(a1)+        * 符号セット
930:
931:            subq.w  #1,d0
932: convstr7:
933:            divu.w  d4,d1
934:            add.b   #$30,d1
935:            move.b  d1,(a1)+
936:            divu.w  d5,d4
937:            clr.w   d1
938:            swap    d1
939:            dbf     d0,convstr7
940: convstr8:
941:            move.w  d6,d7
942:            asl.w   #2,d7
943:            move.l  (a2,d7.w),-(sp)
944:            move.l  #ascii,-(sp)
945:            move.w  handle,-(sp)    * ファイルハンドル
946:            DOS     _WRITE
947:            lea.l   10(sp),sp
948:            cmp.l   (a2,d7.w),d0
949:            bne     write_err       * 総てのデータを書き込めなかった
950:            tst.l   d0
951:            bmi     write_err       * 書き込みエラー
952:            addq.w  #1,d6
953:            cmpi.w  #3,d6
954:            bcs     convstr9
955:            clr.w   d6
956: convstr9:
957:            dbf     d2,x_save@4
958:
959:            move.w  #1,-(sp)        * 現在位置から
960:            move.l  #-1,-(sp)       * ひとつ前
961:            move.w  handle,-(sp)
962:            DOS     _SEEK
963:            addq.l  #8,sp
964:            tst.l   d0
965:            bmi     write_err       * 書き込み(本当はシーク)エラー
966:            move.w  handle,-(sp)    * ファイルハンドル
967:            cmpi.w  #2,d6
968:            beq     convstr9_1
969:            move.w  #$0d,-(sp)
970:            DOS     _FPUTC
971:            addq.l  #2,sp
972:            move.w  #$0a,-(sp)
973:            DOS     _FPUTC
974:            addq.l  #2,sp
975: convstr9_1:
976:            move.w  #$1a,-(sp)      * EOF
977:            DOS     _FPUTC
978:            addq.l  #4,sp
979: convstr10:
980:            move.w  handle,-(sp)
981:            DOS     _CLOSE
982:            addq.l  #2,sp
983:            tst.l   d0
984:            bmi     write_err
985:            clr.l   d0
986:            rts
987: ****************************************
988: * バッファシーク  magic_seek(バッファ番号,オフセット,モード)
989: * モード:0 先頭 1:現在位置 2:ファイルエンド
990: *
991: * エラーならキャリーセット
992: ****************************************
993: x_seek:
994:            tst.w   flush_flg
995:            bmi     flush_err
996:            move.w  14(sp),d0
997:            bsr     chkbuf_no
998:            bcs     para_err        * バッファ番号が無効
999:            bsr     get_bufadrs
1000:           movea.l a0,a6           * パラメータ格納アドレスが置かれたアドレス
1001:           movea.l (a0),a0
1002:           lea.l   buf_use,a1
1003:           move.w  d0,d1
1004:           subq.w  #1,d1
1005:           add.w   d1,d1
1006:           clr.l   d7
1007:           move.w  (a1,d1.w),d7    * バッファ使用量
1008:           lea.l   buf_point,a1
1009:           move.w  34(sp),d2       * モード
1010:           cmpi.w  #3,d2
1011:           bcc     seek_mode_err   * 無効なモードだ
1012:           cmpi.w  #1,d2
1013:           beq     x_seek2         * モードが現在位置だ
1014:           clr.w   (a1,d1.w)       * ポインタを0にする
1015:           bsr     get_paramadrs
1016:           tst.w   d2
1017:           beq     x_seek2         * 先頭
1018:           move.w  d7,(a1,d1.w)    * 最後
1019:           adda.l  d7,a0
1020: x_seek2:
1021:           clr.l   d1
1022:           subq.w  #1,d0
1023:           add.w   d0,d0
1024:           move.w  (a1,d0.w),d1    * D1=ポインタ
1025:           move.l  22(sp),d2
1026:           move.l  d2,d3
```

▶祝氏の復活はないのか。　松田　浩一(19)埼玉県

```
1027:           add.l   #combuf,d2
1028:           bmi     offset_err          * オフセットの値がおかしい
1029:           cmpi.l  #combuf*2,d2
1030:           bhi     offset_err          * オフセットの値がおかしい
1031:           add.l   d3,d1
1032:           bmi     seek_err            * シーク結果<0
1033:           cmp.l   d1,d7
1034:           bcs     seek_err            * シーク結果>使用バッファ
1035:           add.w   d3,(a1,d0.w)        * ワークにポインタを格納
1036:           adda.l  d3,a0
1037:           move.l  a0,(a6)
1038:           lea.l   ret_dat,a0
1039:           move.l  d1,int_dat          * 戻り値(現在のシーク位置)
1040:           clr.l   d0
1041:           rts
1042: ****************************************
1043: * 3個の座標データをチェックする
1044: *
1045: * d0=0,Z=1 d1~d4にパラメータ d4~d6の値は意味なし
1046: * d0≠0,Z=0 座標が不適
1047: ****************************************
1048: get_threeparam
1049:           move.w  #4-1,d0
1050:           bra     get_param
1051: ****************************************
1052: * 4個の座標データをチェックする
1053: *
1054: * d0=0,Z=1 d1~d4にパラメータ d5,d5の値は意味なし
1055: * d0≠0,Z=0 座標が不適
1056: ****************************************
1057: get_fourparam
1058:           move.w  #4-1,d0
1059:           bra     get_param
1060: ****************************************
1061: * 6個の座標データをチェックする
1062: *
1063: * d0=0,Z=1 d1~d6にパラメータ
1064: * d0≠0,Z=0 座標が不適
1065: ****************************************
1066: get_sixparam
1067:           move.w  #6-1,d0
1068: get_param:
1069:           clr.w   d1
1070: get_param2:
1071:           move.l  12+4(sp,d1.w),d2
1072:           add.l   #$8000,d2           * ケタをはかせる
1073:           bmi     zahyou_err
1074:           cmpi.l  #$10000,d2
1075:           bcc     zahyou_err
1076:           add.w   #10,d1
1077:           dbf     d0,get_param2
1078:
1079:           move.w  14+4(sp),d1
1080:           move.w  24+4(sp),d2
1081:           move.w  34+4(sp),d3
1082:           move.w  44+4(sp),d4
1083:           move.w  54+4(sp),d5
1084:           move.w  64+4(sp),d6
1085:           clr.l   d0
1086:           rts
1087: ****************************************
1088: * バッファアドレスを求める
1089: *
1090: * 入力:d0.w バッファ番号
1091: * 出力:a0.l パラメータ格納アドレスが置かれたアドレス
1092: *
1093: * 破壊:a0.l
1094: ****************************************
1095: get_bufadrs:
1096:           lea.l   param_adrs,a0       * パラメータ格納先頭アドレス
1097:           move.w  d0,-(sp)
1098:           subq.w  #1,d0
1099:           asl.w   #2,d0
1100:           adda.w  d0,a0
1101:           move.w  (sp)+,d0
1102:           rts
1103: ****************************************
1104: * パラメータ格納先頭アドレスを求める
1105: *
1106: * 入力:d0.w バッファ番号
1107: * 出力:a0.l パラメータ格納先頭アドレス
1108: *
1109: * 破壊:a0.l
1110: ****************************************
1111: get_paramadrs:
1112:           movem.l d0-d2/a1,-(sp)
1113:           lea.l   param_work,a0
1114:           lea.l   param_format,a1
1115:           subq.w  #1,d0
1116:           beq     get_paramadrs_end   * バッファ番号が1だ
1117:           subq.w  #1,d0
1118:           move.w  d0,d1
1119: get_padrs2:
1120:           clr.l   d2
1121:           move.w  d1,d0
1122:           add.w   d0,d0
1123:           move.w  (a1,d0.w),d2
1124:           adda.l  d2,a0
1125:           dbf     d1,get_padrs2
1126: get_paramadrs_end:
1127:           movem.l (sp)+,d0-d2/a1
1128:           rts
1129: ****************************************
1130: * バッファ番号を調べる
1131: *
1132: * 入力:d0.w バッファ番号
1133: *
1134: * エラーならキャリーセット
1135: *
1136: * 破壊:なし
1137: ****************************************
1138: chkbuf_no:
1139:           tst.w   d0
1140:           beq     chkbuf_no2          * バッファ番号=0
1141:           cmpi.w  #track+1,d0
1142:           bcc     chkbuf_no2          * バッファ番号>track
1143:           andi.b  #%1111_1110,ccr * キャリークリア
1144:           rts
1145: chkbuf_no2:
1146:           ori.b   #%0000_0001,ccr * キャリーセット
1147:           rts
1148: ****************************************
1149: * 指定バッファの空き容量を調べる
1150: *
1151: * 引数:バッファ番号
1152: *
1153: * 戻り値:空き容量
1154: ****************************************
1155: x_free:
1156:           tst.w   flush_flg
1157:           bmi     flush_err
1158:           move.w  14(sp),d0
1159:           bsr     chkbuf_no
1160:           bcs     para_err            * バッファ番号が不適
1161:           lea.l   buf_use,a1
1162:           lea.l   param_format,a2
1163:           subq.w  #1,d0
1164:           add.w   d0,d0
1165:           clr.l   d1
1166:           move.w  (a2,d0.w),d1
1167:           sub.w   (a1,d0.w),d1
1168:           move.l  d1,int_dat
1169:           lea.l   ret_dat,a0
1170:           clr.l   d0
1171:           rts
1172: ****************************************
1173: * バッファにデータが格納可能か調べる
1174: *
1175: * 入力:d0.w バッファ番号
1176: *      d1.w 必要なワード数
1177: *
1178: * 出力:エラーならキャリーセット
1179: *
1180: * 破壊:d1-d4/a4-a6
1181: *
1182: ****************************************
1183: chkbuf:
1184:           lea.l   param_format,a4
1185:           lea.l   buf_point,a5
1186:           lea.l   buf_use,a6
1187:           move.l  d0,-(sp)            * バッファ番号退避
1188:           subq.w  #1,d0
1189:           add.w   d0,d0
1190:           move.w  (a4,d0.w),d2        * バッファ容量
1191:           move.w  (a5,d0.w),d3        * ポインタ
1192:           move.w  (a6,d0.w),d4        * 使用バッファ量
1193:           add.w   d1,d1               * ワード→バイト変換
1194:           add.w   d1,d3               * ポインタを変更
1195:           add.w   d1,d4               * 使用バッファ量を変更
1196:           cmp.w   d2,d4
1197:           bhi     chkbuf_end          * 使用バッファ量>バッファ容量
1198:           move.w  d3,(a5,d0.w)        * ワークに変更したポインタを格納する
1199:           cmp.w   d3,d4
1200:           bhi     chkbuf2             * 使用バッファ量>ポインタ
1201:           move.w  d4,(a6,d0.w)
1202: chkbuf2:
1203:           move.l  (sp)+,d0            * バッファ番号復帰
1204:           andi.b  #%1111_1110,ccr * キャリークリア
1205:           rts
1206: chkbuf_end:
1207:           move.l  (sp)+,d0            * バッファ番号復帰
1208:           ori.b   #%0000_0001,ccr * キャリーセット
1209:           rts
1210: ****************************************
1211: * パレットコードが適正か調べる
1212: *
1213: * d1.w ハレットコード
1214: ****************************************
1215: chk_color:
1216:           move.l  d1.d3               * パレットコード
1217:           move.l  #15,d2
1218:           move.w  #-1,d1
1219:           IOCS    _CRTMOD
1220:           cmpi.w  #8,d0
1221:           bcs     chk_color2          * 16色モードだ
1222:           move.l  #255,d2
1223:           cmpi.w  #12,d0
1224:           bcs     chk_color2          * 256色モードだ
1225:           move.l  #65535,d2
1226:           cmpi.w  #16,d0
1227:           bcs     chk_color2          * 65536色モードだ
1228:           move.l  #15,d2              * 16色モードだ
1229: chk_color2:
1230:           cmp.l   d2,d3
1231:           bhi     color_err
1232:           move.w  d3,d1
1233:           clr.l   d0
1234:           rts
1235: ****************************************
1236: * グラフィック画面が使用できるか
1237: ****************************************
1238: g_chk:
1239:           moveq.l #-1,d1
1240:           IOCS    _APAGE
1241:           tst.l   d0
1242:           rts
1243: ****************************************
1244: *               エラー処理
1245: ****************************************
1246: dim_err:
1247:           moveq.l #1,d0
1248:           lea.l   dim_err_mes,a1
1249:           rts
1250: no_buf_err:
1251:           moveq.l #1,d0
1252:           lea.l   buf_err_mes,a1
1253:           rts
1254: para_err:
1255:           moveq.l #1,d0
1256:           lea.l   para_err_mes,a1
1257:           rts
1258: mode_err:
1259:           moveq.l #1,d0
1260:           lea.l   mode_err_mes,a1
1261:           rts
1262: color_err:
1263:           moveq.l #1,d0
1264:           lea.l   color_err_mes,a1
1265:           rts
1266: zahyou_err:
1267:           moveq.l #1,d0
1268:           lea.l   zahyou_err_mes,a1
1269:           rts
1270: not_com_err:
1271:           moveq.l #1,d0
1272:           lea.l   not_com_mes,a1
1273:           rts
1274: create_err:
1275:           lea.l   ret_dat.a0
1276:           lea.l   create_err_mes,a1
1277:           move.l  #-1,int_dat                     * 戻り値 -1
1278:           rts
1279: write_err:
1280:           lea.l   ret_dat,a0
1281:           lea.l   write_err_mes,a1
1282:           move.l  #-1,int_dat                     * 戻り値 -1
1283:           rts
1284: open_err:
```

▶「皿までどーぞ」から祝氏の信者である僕にとって，ベストライターの３位はショックだった。「祝一平」という名前さえ知らない人がいるなんてありですか？

武藤　智夫(19)東京都

```
1285:           lea.l   ret_dat,a0
1286:           lea.l   open_err_mes,a1
1287:           move.l  #-1,int_dat                  * 戻り値 -1
1288:           rts
1289: read_err:
1290:           lea.l   ret_dat,a0
1291:           lea.l   read_err_mes,a1
1292:           move.l  #-1,int_dat                  * 戻り値 -1
1293:           rts
1294: zahyou_err2:
1295:           moveq.l #1,d0
1296:           lea.l   zahyou_err_mes2,a1
1297:           rts
1298: mukou_err:
1299:           moveq.l #1,d0
1300:           lea.l   mukou_err_mes,a1
1301:           rts
1302: offset_err:
1303:           moveq.l #1,d0
1304:           lea.l   ret_dat,a0
1305:           lea.l   offset_err_mes,a1
1306:           move.l  #-1,int_dat                  * 戻り値 -1
1307:           rts
1308: seek_err:
1309:           moveq.l #1,d0
1310:           lea.l   ret_dat,a0
1311:           lea.l   seek_err_mes,a1
1312:           move.l  #-1,int_dat                  * 戻り値 -1
1313:           rts
1314: g_err:
1315:           moveq.l #1,d0
1316:           lea.l   ret_dat,a0
1317:           lea.l   g_err_mes,a1
1318:           move.l  #-1,int_dat                  * 戻り値 -1
1319:           rts
1320: init_err:
1321:           moveq.l #1,d0
1322:           lea.l   init_err_mes,a1
1323:           rts
1324: flush_err:
1325:           moveq.l #1,d0
1326:           lea.l   flush_err_mes,a1
1327:           rts
1328: bufend_err:
1329:           moveq.l #1,d0
1330:           lea.l   bufend_err_mes,a1
1331:           rts
1332: seek_mode_err:
1333:           moveq.l #1,d0
1334:           lea.l   seek_mode_mes,a1
1335:           rts
1336: ****************************************
1337: * インフォメーションテーブル
1338: ****************************************
1339: x_init:
1340: x_run:
1341: x_end:
1342: x_sys:
1343: x_brk:
1344: x_ctrl_d:
1345: x_res1:
1346: x_res2:
1347:           rts
1348:
1349: ptr_token:
1350:           dc.b    'magic_line',0
1351:           dc.b    'magic_connect',0
1352:           dc.b    'magic_spline',0
1353:           dc.b    'magic_box',0
1354:           dc.b    'magic_triangle',0
1355:           dc.b    'magic_fill',0
1356:           dc.b    'magic_circle',0
1357:           dc.b    'magic_window',0
1358:           dc.b    'magic_mode',0
1359:           dc.b    'magic_wipe',0
1360:           dc.b    'magic_para',0
1361:           dc.b    'magic_data',0
1362:           dc.b    'magic_pers',0
1363:           dc.b    'magic_disp',0
1364:           dc.b    'magic_color',0
1365:           dc.b    'magic_screen',0
1366:           dc.b    'magic_init',0
1367:           dc.b    'magic_auto',0
1368:           dc.b    'magic_flush',0
1369:           dc.b    'magic_free',0
1370:           dc.b    'magic_putbuf',0
1371:           dc.b    'magic_getbuf',0
1372:           dc.b    'magic_save',0
1373:           dc.b    'magic_load',0
1374:           dc.b    'magic_savea',0
1375:           dc.b    'magic_seek',0
1376:           dc.b    'magic_buffer',0
1377:           dc.b    'magic_bufon',0
1378:           dc.b    'magic_bufoff',0
1379:           dc.b    0
1380:           .even
1381: ptr_param:
1382:           dc.l    line_par
1383:           dc.l    connect_par
1384:           dc.l    spline_par
1385:           dc.l    box_par
1386:           dc.l    triangle_par
1387:           dc.l    boxfull_par
1388:           dc.l    circle_par
1389:           dc.l    window_par
1390:           dc.l    mode_par
1391:           dc.l    wipe_par
1392:           dc.l    para_par
1393:           dc.l    data_par
1394:           dc.l    pers_par
1395:           dc.l    disp_par
1396:           dc.l    color_par
1397:           dc.l    screen_par
1398:           dc.l    init_par
1399:           dc.l    auto_par
1400:           dc.l    flush_par
1401:           dc.l    free_par
1402:           dc.l    putbuf_par
1403:           dc.l    getbuf_par
1404:           dc.l    save_par
1405:           dc.l    load_par
1406:           dc.l    save@_par
1407:           dc.l    seek_par
1408:           dc.l    buffer_par
1409:           dc.l    bufon_par
1410:           dc.l    bufoff_par
1411: line_par:
1412:           dc.w    int_val
1413:           dc.w    int_val
1414:           dc.w    int_val
1415:           dc.w    int_val
1416:           dc.w    int_val
1417:           dc.w    void_ret
1418: connect_par:
1419:           dc.w    aryl_i
1420:           dc.w    void_ret
1421: spline_par:
1422:           dc.w    int_val
1423:           dc.w    int_val
1424:           dc.w    int_val
1425:           dc.w    int_val
1426:           dc.w    int_val
1427:           dc.w    int_val
1428:           dc.w    int_val
1429:           dc.w    void_ret
1430: box_par:
1431:           dc.w    int_val
1432:           dc.w    int_val
1433:           dc.w    int_val
1434:           dc.w    int_val
1435:           dc.w    int_val
1436:           dc.w    void_ret
1437: triangle_par:
1438:           dc.w    int_val
1439:           dc.w    int_val
1440:           dc.w    int_val
1441:           dc.w    int_val
1442:           dc.w    int_val
1443:           dc.w    int_val
1444:           dc.w    int_val
1445:           dc.w    void_ret
1446: boxfull_par:
1447:           dc.w    int_val
1448:           dc.w    int_val
1449:           dc.w    int_val
1450:           dc.w    int_val
1451:           dc.w    int_val
1452:           dc.w    void_ret
1453: circle_par:
1454:           dc.w    int_val
1455:           dc.w    int_val
1456:           dc.w    int_val
1457:           dc.w    int_val
1458:           dc.w    void_ret
1459: window_par:
1460:           dc.w    int_val
1461:           dc.w    int_val
1462:           dc.w    int_val
1463:           dc.w    int_val
1464:           dc.w    void_ret
1465: mode_par:
1466:           dc.w    char_val
1467:           dc.w    void_ret
1468: wipe_par:
1469:           dc.w    void_ret
1470: para_par:
1471:           dc.w    int_val
1472:           dc.w    int_val
1473:           dc.w    void_ret
1474: data_par:
1475:           dc.w    char_val
1476:           dc.w    void_ret
1477: pers_par:
1478:           dc.w    void_ret
1479: disp_par:
1480:           dc.w    void_ret
1481: color_par:
1482:           dc.w    int_val
1483:           dc.w    void_ret
1484: screen_par:
1485:           dc.w    char_val
1486:           dc.w    void_ret
1487: init_par:
1488:           dc.w    void_ret
1489: auto_par:
1490:           dc.w    char_val
1491:           dc.w    int_ret
1492: flush_par:
1493:           dc.w    char_omt
1494:           dc.w    void_ret
1495: free_par:
1496:           dc.w    char_val
1497:           dc.w    int_ret
1498: putbuf_par:
1499:           dc.w    char_val
1500:           dc.w    aryl_i
1501:           dc.w    int_ret
1502: getbuf_par:
1503:           dc.w    char_val
1504:           dc.w    int_ret
1505: save_par:
1506:           dc.w    char_val
1507:           dc.w    str_val
1508:           dc.w    int_omt
1509:           dc.w    int_omt
1510:           dc.w    void_ret
1511: load_par:
1512:           dc.w    char_val
1513:           dc.w    str_val
1514:           dc.w    int_ret
1515: save@_par:
1516:           dc.w    char_val
1517:           dc.w    str_val
1518:           dc.w    int_omt
1519:           dc.w    int_omt
1520:           dc.w    void_ret
1521: seek_par:
1522:           dc.w    char_val
1523:           dc.w    int_val
1524:           dc.w    char_val
1525:           dc.w    int_ret
1526: buffer_par:
1527:           dc.w    char_val
1528:           dc.w    void_ret
1529: bufon_par:
1530:           dc.w    void_ret
1531: bufoff_par:
1532:           dc.w    void_ret
1533: ptr_exec:
1534:           dc.l    x_line
1535:           dc.l    x_connect
1536:           dc.l    x_spline
1537:           dc.l    x_box
1538:           dc.l    x_triangle
1539:           dc.l    x_boxfull
1540:           dc.l    x_circle
1541:           dc.l    x_window
1542:           dc.l    x_mode
```

▶苦節5年，やっとX68000XVIを買いました。初心者には81Mバイトは果てしないです(少し自慢)。
大場　幹夫(22)東京都

```
1543:           dc.l     x_wipe
1544:           dc.l     x_para
1545:           dc.l     x_data
1546:           dc.l     x_pers
1547:           dc.l     x_disp_flame
1548:           dc.l     x_color
1549:           dc.l     x_screen
1550:           dc.l     x_minit
1551:           dc.l     x_auto
1552:           dc.l     x_flush
1553:           dc.l     x_free
1554:           dc.l     x_putbuf
1555:           dc.l     x_getbuf
1556:           dc.l     x_save
1557:           dc.l     x_load
1558:           dc.l     x_save@
1559:           dc.l     x_seek
1560:           dc.l     x_buffer
1561:           dc.l     x_bufon
1562:           dc.l     x_bufoff
1563:
1564:           .data
1565:
1566: dim_err_mes:
1567:           dc.b     '変数の型が違います',0,0
1568: buf_err_mes:
1569:           dc.b     'バッファがいっぱいです',0,0
1570: para_err_mes:
1571:           dc.b     '無効なバッファ番号です',0,0
1572: zahyou_err_mes:
1573:           dc.b     '無効な座標を指定しました',0,0
1574: mode_err_mes:
1575:           dc.b     '無効なラインモードを指定しました',0,0
1576: color_err_mes:
1577:           dc.b     '無効なパレットコードを指定しました',0,0
1578: mukou_err_mes:
1579:           dc.b     '無効な設定データです',0,0
1580: not_com_mes:
1581:           dc.b     'コマンドではありません',0,0
1582: create_err_mes:
1583:           dc.b     'ファイルが作れません',0,0
1584: write_err_mes:
1585:           dc.b     'ファイルの書き込みに失敗しました',0,0
1586: open_err_mes:
1587:           dc.b     'ファイルがオープンできません',0,0
1588: read_err_mes:
1589:           dc.b     'ファイルの読み込みに失敗しました',0,0
1590: zahyou_err_mes2:
1591:           dc.b     '現在のモードでは設定できません',0,0
1592: offset_err_mes:
1593:           dc.b     '無効なオフセットです',0,0
1594: seek_err_mes:
1595:           dc.b     '未使用バッファにポインタが移動しました',0,0
1596: g_err_mes:
1597:           dc.b     'グラフィック画面が初期化されていません',0,0
1598: init_err_mes:
1599:           dc.b     '3D用ワークが初期化されていません',0,0
1600: flush_err_mes:
1601:           dc.b     'バッファが初期化されていません',0,0
1602: bufend_err_mes:
1603:           dc.b     'ハッファエンドです',0,0
1604: seek_mode_mes:
1605:           dc.b     '無効なモードです',0,0
1606:
1607: buf_no:
1608:           dc.w     1                      * デフォルトバッファ番号
1609: handle:
1610:           ds.w     1                      * ファイルハンドル
1611: ascii:
1612:           ds.b     7                      * 文字列格納領域
1613:           dc.b     ','
1614:           dc.b     13,10
1615: format:
1616:           dc.l     8,8,10
1617:
1618: ret_dat:
1619:           dc.w     0
1620:           dc.l     0
1621: int_dat:
1622:           dc.l     0
1623: init_flg:
1624:           dc.w     -1                     * 3D用バッファが初期化されていない
1625: flush_flg:
1626:           dc.w     -1                     * バッファが初期化されていない
1627: buf_flg:
1628:           dc.w     -1                     * バッファに格納しない
1629: param_point:
1630:           dc.l     param_work             * パラメータを格納先頭アドレス
1631: param_format:
1632:           dc.w     combuf
1633:           dc.w     bufsize
1634:           dc.w     bufsize
1635:           dc.w     bufsize
1636:           dc.w     bufsize
1637:
1638:           .bss
1639:
1640: buf_use:
1641:           ds.w     track                  * 各バッファの使用容量
1642: buf_point:
1643:           ds.w     track                  * 各バッファのポインタ
1644: param_work:
1645:           ds.b     combuf
1646:           ds.b     bufsize*(track-1)      * パラメータを格納する領域
1647: param_adrs:
1648:           ds.l     track                  * パラメータを格納するアドレス
1649: parameter:
1650:           ds.w     bufsize
1651:
1652:           .end
1653:
```

## リスト5

```
 1: *
 2: * mode.s           version 1.01
 3: *
 4: * 91/ 5/ 1         IOCSコールを使ったXORモードを追加
 5: *
 6:
 7: _DRAWMODE          equu     $B0
 8:
 9:         .include           iocscall.mac
10:         .include           work.h
11:
12:         .xdef              mode
13:         .xdef              disp_mode
14:
15:         .text
16:         .even
17:
18: mode:
19:         move.w  (a0)+,d1
20:         move.w  d1,disp_mode       * ラインモード
21:         subq.w  #1,d1
22:         beq     xor                * 1
23:         subq.w  #1,d1
24:         beq     pset               * 2
25:         rts
26: pset:
27:         moveq.l #0,d1
28:         IOCS    _DRAWMODE
29:         rts
30: xor:
31:         moveq.l #1,d1
32:         IOCS    _DRAWMODE
33:         rts
34:
35:         .data
36:
37: disp_mode:
38:         ds.w    1
39:
40:         .end
41:
```

## リスト6

```
  1: *
  2: * disp_flame.s ラインモード(XOR,ORのみ)対応
  3: *
  4:
  5:         .include           iocscall.mac
  6:         .include           work.h
  7:
  8:         .xdef              disp_flame
  9:
 10: disp_flame:
 11:         move.l  a0,-(sp)               * a0退避
 12:         move.l  #lin_buf,d1
 13:         move.l  #lin_buf2,d2
 14:         move.l  #disp_buf,d3
 15:         move.l  #disp_buf2,d4
 16:         move.b  page,d5                * ページ
 17:         lea.l   line_data,a1
 18:         lea.l   pct,a0
 19:         lea.l   pct2,a2
 20:         lea.l   lct,a4
 21:         lea.l   lct2,a6
 22:         tst.b   page
 23:         beq     disp_flame2            * ページ0だ
 24:         exg     d1,d2                  * ページ1の時
 25:         exg     d3,d4
 26:         exg     a0,a2
 27:         exg     a4,a6
 28: disp_flame2:
 29:         clr.w   (a2)                   * 裏画面のpctをクリア
 30:
 31:         movea.l d1,a2                  * 線分バッファ
 32:         movea.l d3,a3                  * 2D変換後のデータバッファ
 33:
 34:         eori.b  #1,d5
 35:         move.b  d5,page
 36:         move.b  d5,d1
 37:         IOCS    _APAGE                 * ページ反転
 38:
 39:         move.w  (a4),d7
 40:         beq     disp_flame4            * 線分が無い
 41:         subq.w  #1,d7
 42:         move.w  d7,a5                  * d7をA5に退避
 43:         move.l  a2,d3                  * A2をD3に複写
 44: disp_flame3:
 45:         move.w  (a2)+,d1
 46:         add.w   d1,d1
 47:         add.w   d1,d1
 48:         move.l  (a3,d1.w),(a1)
 49:         move.w  (a2)+,d1
 50:         add.w   d1,d1
 51:         add.w   d1.d1
 52:         move.l  (a3,d1.w),4(a1)
 53:         IOCS    _LINE
 54:         dbf     d7,disp_flame3
 55: disp_flame4:
 56:         move.b  d5,d1
 57:         addq.w  #1,d1
 58:         IOCS    _VPAGE
 59:         move.b  d5,d1
 60:         eori.b  #1,d1
 61:         IOCS    _APAGE
 62:
 63:         moveq.l #-1,d1
 64:         IOCS    _CRTMOD
 65:         cmpi.w  #4,d0
 66:         bcs     disp_flame_1024
 67:         cmpi.w  #12.d0
 68:         bcs     disp_flame_512
 69: disp_flame_1024:                       * 1ライン下が +1024
 70:         move.w  a5,d7
 71:         movea.l d3,a2
 72:         bra     disp_flame5
 73: disp_flame_512                         * 1ライン下が +512
 74:         move.w  (a6),d7
 75:         beq     disp_flame_end         * 線分が無い
 76:         subq.w  #1,d7
 77:         movea.l d2,a2                  * 線分バッファ
 78:         movea.l d4,a3                  * 2D変換後のデータバッファ
 79: disp_flame5:
 80:         move.w  line_data+8,d5
 81:         clr.w   line_data+8
 82: disp_flame6:
 83:         move.w  (a2)+,d1
 84:         add.w   d1,d1
 85:         add.w   d1,d1
 86:         move.l  (a3,d1.w),(a1)
 87:         move.w  (a2)+,d1
 88:         add.w   d1,d1
 89:         add.w   d1,d1
 90:         move.l  (a3,d1.w),4(a1)
 91:         IOCS    _LINE
 92:         dbf     d7,disp_flame6
 93:         move.w  d5,line_data+8
 94: disp_flame_end:
 95:         move.l  d2,line_adr            * 線分バッファ
 96:         move.l  d4,point_adr           * 2D変換後のデータバッファ
 97:         clr.w   (a6)                   * 裏画面のlctをクリア
 98:         move.l  (sp)+,a0
 99:         rts
100:
101:         .end
102:
```

▶やっと2Mバイトに増設したが，まだ“狭い部屋”のうちなんですね。

出倉　義明(20)神奈川県

# tinyCalcの関数を作る

tinyCalcは5月号付録ディスクに収録した表計算ソフトです。今回は応用編として自分独自の関数を拡張するために必要な情報を掲載します。ソースリストつきですからC言語の使える方なら拡張は難しくないでしょう。カスタマイズして使いやすいツールに仕上げてください。

Izumi Daisuke
泉 大介

さっそく先月のおさらいを兼ねて例題から始めましょう。数値演算の応用例を2つ紹介します。

**●tinyCalcで定積分**

次の関数,

$f(x)=x^3-6x^2+9x$

を不定積分すると,

$$F(x)=\frac{1}{4}x^4-2x^3+\frac{9}{2}x^2+C$$

となります。F(x)を微分するとf(x)になるように係数を調整すればいいだけですから,ここまでは比較的簡単。私も好きでした。ところが,

$$\int_1^5 x^3-6x^2+9x\,dx$$

となると事態は変わってきます。

$$\left[-\frac{1}{4}x^4-2x^3+\frac{9}{2}x^2\right]_1^5$$

という極悪非道な数値計算を,しかも手で,やらなければならなくなってしまうからです。これをtinyCalcでなんとかしようというのが狙いです。

まずA1セルに,

@pow([0,0],4)/4－2＊@pow([0,0],3)+9＊@pow([0,0],2)/2

という式を書き込みます。これは上の[ ]の中身です。xの代わりに自分自身のセルに入っている値を使おうというわけです。次にメニューの「複写」を使って,この式をB1セルにコピーします。

おもむろにC1セルに,

a1＝5;
b1＝1;
@fnc(a1)－@fnc(b1)

と書き込めば,面倒な計算はすべてtinyCalcがやってくれ,C1セルには答えが表示されます。式の入っているA1セルに,

a1＝5

などとやって数値をセットしても大丈夫なのかという不安があるかと思いますが,どっこいこれがOKなのです(だってそう作ったんだもん)。

## 自分専用の関数を追加する

tinyCalcには面白そうな関数を集めてみましたが,皆さんの用途によってはもっと別の関数がほしいと思われるかもしれません。たとえば範囲の合計を求めるのではなく,範囲内のデータをすべて掛けた値がほしい場合もあるでしょう。自分専用の関数を追加するために必要な情報をここにまとめておきます。

一般関数はuserfunc.cにまとめてあります。関数を追加する場合はこのファイルを変更してください。16～37行は,このファイルで定義されている関数のプロトタイプ宣言です。プロトタイプ宣言をご覧になってわかるように,定義できる関数はdouble(倍精度実数)を返す関数に限られます。46～78行は,tinyCalcで使う関数名と,実際に呼び出す関数名をペアにしたものを収める配列functionsです。自分で関数を追加した場合には,忘れずにここに関数名を登録するようにしてください。

**●大域変数**

自作関数を作成する場合に注意したい変数は2つ。8行のcol変数と11行のPARSE変数です。

col変数は,皆さんがセルに書き込んだデータを示すポインタです。このポインタの示すアドレスに,セルに書き込まれたデータが入っています。

PARSE変数は現在式を実行しているのか,それとも文法をチェックしているだけなのかを示す変数です。文法をチェックしているときにはPARSE変数は1になっています。

**●エラー発生時の処理**

式が正しくないなどのエラーが発生したときは,

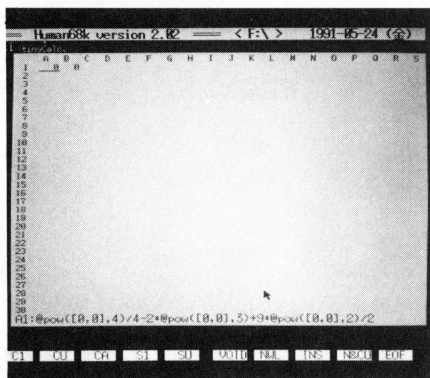
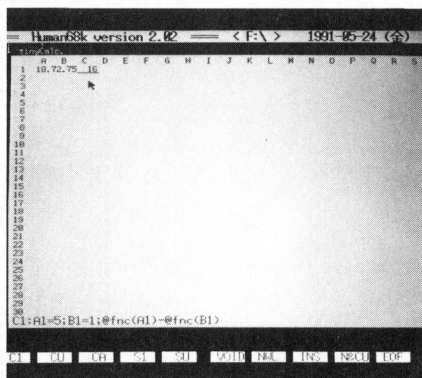
longjmp( jbuf, エラー番号 );

を実行してください。エラー番号とその種類を挙げておきます。

1) 式の間違い
2) 扱おうとするセルがセル範囲を越えた
3) 規定外の値を使った
6) メモリがなくなった

7以上のエラー番号を指定すると,エラーメッセージではなくエラー番号が表示されます。エラー番号2はほとんど指定することはないでしょう。なぜなら,ユーザーがチェックしなくても,データを書き込む関数を利用すればそちらでやってくれるからです。エラー番号4,5は行列用のエラーメッセージなので省略しました。

**●関数の作り方1**

最初に簡単な例として,引数がひとつだけの場合を例に説明しましょう。リスト1をご覧ください。これは,userfunc.cファイルの215行にあるmysin関数です。ユーザー

この式が……

こうなる

関数にきた時点で，colは関数名の後ろのカッコのあとを指しています。つまり，

@sin(■10 )

の位置です。sin関数は式を引数に取ります。式はsetupという関数を呼び出せば自動的に評価されるようになっています。setup関数は，演算子のなかでももっとも優先順位の低い「代入」を処理する関数で，代入より優先順位の高いすべての式を評価することができます。5行ではこのsetup関数を呼び出して引数を取り出しています。

引数が取り出されたあと，colは取り出した引数のあとを指しています。つまり，

@sin( 10■)

の位置です。6行では引数の後ろにスペースが入っている場合を考え，spaceCutで空白を飛ばしています。これでcolは')'を指すようになったはずです。7行でそれをチェックし，カッコが閉じていれば8行です。colがカッコの次を指すように1つ大きくし，sinの値を計算してそれを返します。

カッコが閉じていなければ「文法エラー」です。このときは11行のように処理してください。

複数の引数を取る関数を作る場合も手順は同様です。リスト1で')'のチェックを行っている部分を引数の区切りである','のチェックを行うように変更します。OKならcolを1つ大きくしてsetupで次の引数を取り出す。この作業を続けていけばいいわけです。必要な引数が揃ったら，カッコが閉じているかどうかのチェックを行い，あとはリスト1と同様です。

tinyCalcの関数は，このように文法チェックと関数の実行を同時に行うように書く必要があります。皆さんが必要とするであろう関数はリスト1のタイプのものがほとんどだと思います。

**●関数の作り方2**

相対セル指定はコピーした先でもセル位置を相対的に指定でき便利なのですが，自分の操作したいセルを明示的に指定することが困難だという問題点もあります。

このような要求に応える関数の例がリスト2です。この関数は，現在評価中のセルと，ユーザーが明示的に指定したセルとの相対座標を取り出す関数です。たとえばB1セルに，

@x(a1)

と書き込めば，現在評価中のセルであるB1と，指定されたセルであるA1の列方向の相対座標－1が表示されます。tinyCalcで，B1セルからA1セルを相対指定するには

[－1,0]

としなければなりませんが，この関数を使えば，

[@x(a1),0]

と明示的に指定できるようになります。

リスト2をご覧ください。1行で宣言している変数は，現在評価中のセルのX座標を保持している変数です。ここには挙げていませんが，現在評価中のセルのY座標を保持している変数もあり，こちらにはevalYという名前がついています。それぞれの変数は，

$1 \leqq \text{evalX} \leqq 23$

$1 \leqq \text{evalY} \leqq 66$

の範囲の値を取ります。現在評価中のセルがB1の場合，evalXは2，evalYは1になります。

tinyCalcの@x関数には，locxというC言語の関数名をつけました。引数が式の場合はリスト1のようにsetup関数を呼び出して取り出せばいいのですが，セル指定を取り出す場合には7行のようにsetparam関数を使います。これで引数として指定されたセルの座標がx，yに取り出されます。setparam関数は，絶対セル指定，相対セル指定のいずれのセル指定が使われていても，必ず絶対座標を取り出します。取り出されたxとevalXの差を取れば，それが列方向の相対座標となります。

実際にこの関数をuserfunc.cに組み込むには，ファイルの一番最後にリスト2のプログラムを書き込み，関数のプロトタイプ宣言，

double locx( void );

をプロトタイプ宣言部に，tinyCalcの関数名とC言語の関数名のペア，

{ "x", locx }

をfunctions配列の宣言部の最後につけ加えてください。

@x関数だけでは片手落ちです。当然@y関数もサポートしなければなりません。evalY変数を，

extern int evalY;

と宣言することと，関数が返す値が

return( y－evalY );

となること以外はリスト2と同じです。自分で追加してみてください。

**●副作用のある関数の作り方**

関数の中には値をreturnで返すだけでなく，tinyCalcが扱っている表そのものに影響を与えるものがあります。たとえば@fill関数は，指定された範囲にデータを書き込みます。これは通常の関数のように単純に値を返すだけでは実現できません。

数値をセルに直接書き込むには，setnum関数を使います。setnum関数は，

setnum( X座標，Y座標，数値 )；

として使います。X，Y座標は絶対位置です。つまり，A1セルならX＝Y＝1ですし，B1セルならX＝2，Y＝1となります。書き込むセルに式がセットされていても，その式はクリアされません。セルに入力された式とセルの値は，内部で別々に管理されているからです。なお，setnum関数で数値をセルにセットしても，画面には表示されません。もし画面に表示したい場合はprntCell関数を利用してください。この関

**リスト1 mysin関数**

```
 1: double mysin( void )
 2: {
 3:     double  x;
 4:
 5:     x = setup();                ← xに引数を取り出す
 6:     spaceCut();                 ← 空白をカットして
 7:     if ( *col == ')' ) {        ← カッコが閉じているか確認
 8:         col++;                  ← OKならカッコの次の文字を指す
 9:         return( sin( x ));      ← sinを計算してそれを返す
10:     } else
11:         longjmp( jbuf, 1 );     ← 文法エラー
12: }
```

**リスト2 @X関数を作る**

```
 1: extern int evalX;               ← 現在評価中のセル
 2:
 3: double locx( void )
 4: {
 5:     int     x, y;
 6:
 7:     setparam( &x, &y );         ← セル指定を数値に直す
 8:     spaceCut();                 ← 空白をカットして
 9:     if ( *col == ')' ) {        ← カッコが閉じているか確認
10:         col++;
11:         return( x-evalX );      ← 相対座標を得て返す
12:     } else
13:         longjmp( jbuf,1 );      ← 文法エラー
14: }
```

数は,

prntCell( X座標, Y座標 );

として使います。

さて,setnum関数を使う際にはこれまで説明してきた関数には不要だった注意が必要です。それは,「文法チェックをしている最中にはデータを書き込まないようにする」という点です。もし@fill関数でこの注意が忘れられると,

@fill(a1,j10,1,100))

などと誤った式(閉じカッコが多い)を書いてしまった場合,文法ミスを指摘される前にA1-J10セルにデータが埋め込まれてしまうことになります。

このようなことが起きないように,

```
if ( PARSE )
    return( 0 );
```

という行を必ず書き入れ,文法チェックはこの行の前で,setnum関数の使用はこの行のあとで行うようにしてください。@fill関数が参考になるかと思います。

**●その他こまごまとしたサポート関数**

setparam関数はセル指定を1個取り出すための関数です。範囲指定をする関数では範囲の最初と最後の2つのセル指定を取り出さなければなりません。これをサポートするのがset2param関数です。使い方は,

```
int xs, xe, ys, ye;
set2param( &xs, &xe, &ys, &ye );
```

とします。これで最初のセル指定がxsとysに,次のセル指定がxeとyeに取り出されます。絶対,相対セル指定の区別なく,取り出されるのは絶対座標です。

2つのセル指定を引数に取る関数は,範囲を計算対象とする関数です。このとき注意しなければならないのは,セル幅が変更されている場合があるということです。MAT関数のようにセル幅を無視して計算するのか,@fill関数のようにセル幅に対応してデータをセットするのかは,皆さんが決めてください。セル幅に対応した関数を作るのなら,countup関数が有効です。この関数は,

countup( X座標 )

のように使います。これで,画面に表示されている右隣のセルに移動するには,X座標をいくつ増やせばいいかが返されます。

for (i=1; i<=23; i+=countup(i))

のように使えばいいでしょう。@fill関数の中に例があります。

@fill関数では,ここまでに説明していない大域変数を参照しています。editMode変数がそれで,これは領域へのデータ入力で,縦方向優先入力になっているか,横方向優先入力になっているかを保持している変数です。ご参考までに。

## 終わりに

tinyCalcをしばらく使っているうちに,@fnc関数の仕様の不備に気づきました。階乗を再帰で行うトリックをひねり出したので満足して気づかなかったのでしょう。定積分の説明に,

a1=5;

b1=1;

@fnc(a1)−@fnc(b1)

という式を書いているところがあります。単式の中に連式を使えないため,本当は

(a1=5;@fnc(a1))−(a1=1;@fnc(a1))

と書きたかったところを,わざわざこのようにしたのです。このままでは@fnc関数が半分死んでいるのも同然です。

@fnc( a1, 5 )

と書けるようにしておくべきでした。これは,関数呼び出しを行うセルに2番目の引数をあらかじめセットしたあと,セルの実行を開始する,という意味です。これなら入力した式を別のセルにコピーする必要もなくなります。

そこで@pfncという新しい関数を用意してみました。セルの実行前にデータをセットする機能を持った@fnc関数です。リスト3のようになります。この関数は諸般の事情により,matfunc.cファイルの最後に付け加えてください。また,皆さんには使っていただきたくない綱渡り的な機能を使っているため,あえて説明はしません。リスト3の入力が終わったら,userfunc.cのプロトタイプ宣言に,

double pfnc( void );

の1行を付け加え,さらにfunctions配列の初期化部分に,

{ "pfnc", pfnc }

1行を付け加えて再コンパイルしてください。これで先の定積分の式は,A1セルに式をセットしたあと,A2セルに,

@pfnc(a1, 5)−@pfnc(a1,1)

と書き込むだけでスマートに記述できるようになります。

自分の必要な関数を追加して,どうぞtinyCalcライフをエンジョイしてください。

### 掟破りの@fnc

@fncを使って,こんなことも可能です。A1セルに次の式を書き込んでみてください。

```
#if (a1==0)
  1
#else
  a1=a1-1;
  (a1+1) *@fnc (a1)
#endif
```

そしてB1セルに,

```
a1=5;
#gosub a1
```

と書き込めば,5の階乗を計算することができます。通常,階乗はループを使って計算するものなのですが,ここでは「再帰」を使ってみました。こんなことができるのは,tiny Calcが式を前から順に計算していくためです。(a1+1)の計算が終わった時点で,計算結果はC言語の変数に格納されています。したがってそのあとで@fnc (a1) を計算しても支障ないのです。仮に,

@fnc (a1) * (a1+1)

と書き換えたら,とんでもない値がセットされます。

XCのライブラリにはバグがあるようで,再帰が深くなりすぎると暴走します。無謀な再帰はしないでください。

**リスト3 @pfncの作成**

```
 1: double pfnc( void )
 2: {
 3:     int      x, y, a, xb, yb;
 4:     char     *c;
 5:     jmp_buf backup;
 6:
 7:     setparam( &x, &y );
 8:     spaceCut();
 9:     if ( *col != ',' )
10:         longjmp( jbuf, 1 );
11:     col++;
12:     a = setup();
13:     spaceCut();
14:     if ( *col != ')' )
15:         longjmp( jbuf, 1 );
16:     col++;
17:     if ( PARSE )
18:         return( 0 );
19:
20:     c = col;
21:     xb = evalX; yb = evalY;
22:     backup[ 0 ] = recalbuf[ 0 ];
23:     setnum( x, y, a );
24:     recalc( x, y );
25:     recalbuf[ 0 ] = backup[ 0 ];
26:     evalX = xb; evalY = yb;
27:     col = c;
28:     return( CalcTable[ x ][ y ].val );
29: }
```

よいこのSX-WINDOW講座（第3回）

# ダイアログで対話する（後編）

Nakamori Akira 中森　章

**前回に続いてダイアログウィンドウの扱い方を考えていきましょう。今回は現在のダイアログウィンドウでは具合の悪い部分の対処のしかたや，ダイアログマネージャでは実現できない複雑な対話を行うための手法も考えてみます。**

今回は，前回のダイアログの続きです。前回はダイアログウィンドウの基本的な操作を学びました。今回はもっと簡単にダイアログを扱う方法と，もっと複雑だけどより柔軟な対話を行うためのダイアログについて学ぶことにします。

なお，今回のプログラム例ではダイアログを行う関数を呼び出す側のスケルトンプログラムは掲載していません。ダイアログを行うサブルーチンはどのような（OBJC型の）スケルトンプログラムからも呼び出すことができるので，スケルトンプログラムを一意に決める必要はないと思ったからです（前回のスケルトンプログラムを打ち込んでしまった人には不要なリストになりますしね）。どうしても必要な人は前回（6月号）の連載を見てください。

## 複数のテキストを持つダイアログ

現在のSX-WINDOWではダイアログウィンドウ上に2つ以上の編集可能テキストがあると入力した文字列を正しく取り出すことができません（前回を参照のこと）。これをそのままにしていたのでは面白くないので，なんとか工夫をして複数の編集可能テキストが扱えるようにしてみましょう[1]。

このとき頼りになるのがフィルタ関数です。現在の最大の問題点は編集対象とするテキストを切り替えたときにこれまで編集していた文字列が消えてしまい，最後に編集していた文字列しか残らない（ことがある）ということです。そこで，現在編集中となっている編集可能テキストを常に監視しておいて，マウスによって別の編集可能テキストが選択された場合は，これまでの編集結果を別の領域に退避しておくという処理を行うことにします。こうすれば，最後に編集対象になっていた編集可能テキスト以外の編集可能テキストの内容（文字列）はその退避場所から持ってきて参照することができますね。これはフィルタ関数の中で簡単に行うことのできる処理なのです。

リスト1に2つの編集可能テキストを持つダイアログの例を示します。前回紹介したプログラム例で使用したフィルタ関数と比べると，フィルタ関数の中にマウス左ボタンダウン時の処理が追加されているのがわかると思います。編集対象のテキストが切り替わるのを発見したとき，ダイアログ用のDITGet関数は信用できないので，テキストマンの関数である，

　TMGetText

という関数を使用して文字列を取り出すことにします。このとき，ダイアログウィンドウの中で現在編集対象となっているテキストへのハンドル（tEdit型へのハンドル）が必要になります。そのハンドルはdialogという構造体のdTextフィールドに格納されていますから，それをTMGetTextに渡して文字列を得るようにしています[2]。

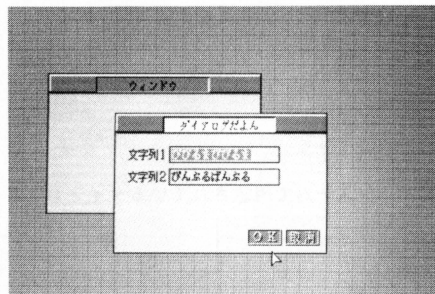
このように，フィルタ関数を使用すればダイアログの操作に広がりを持たせることができます。余力のある人は，もともとの仕様のようにTABキーで編集対象のテキストを切り替えるという処理を追加してみてください。

---

1）　現在編集対象となっている編集可能テキストはマウスカーソルで選択する（か，あるいはタブを入力する）ことで切り替えることができる。このとき，直前に編集していた文字列の内容が新しいテキストにコピーされるようになっている。

2）　dialogという構造体はsxlib.h(sxdef.h)内で定義されている。DMOpenはこのdialog型へのポインタを返す。

## リソースを利用するダイアログ

ダイアログウィンドウはダイアログアイテムテーブルを用意してDMOpen関数を呼び出せば簡単に作成することができます。しかし，世の中にはもっと楽をしたい人もいるはずです。ダイアログアイテムリストを作るときは制御ボタンの位置などを考えなければなりません。これは結構面倒なことです。そこで，制御ボタンの配置などであれこれ頭を悩ますよりは，あらかじめ汎用的なダイアログウィンドウの雛形を作っておき，必要に応じてそれに変更を加えて使うという方法が考えられます。それがリソースを用いる方法です。

**リスト1の実行結果**

リソースとしてのダイアログウィンドウはのっぺらぼうのウィンドウに制御ボタンが並んだだけのものです。ユーザーはこれに対して用途別に文字列を書き込んで専用のダイアログウィンドウに仕立て上げるのです。すなわち，文字列をダイアログアイテムの一部として登録してしまうのではなく，表示されたウィンドウに対して，あとから重ね書きをするような感覚です。この方法の利点は，文字の描画にダイアログマンではなくグラフマンの関数を利用するので，文字に影をつけたり，字体を変更したりなどの操作が自由にできるということでしょう。

リソースはユーザーが自由に作ることもできますが，SX-WINDOW自体にもシステムが利用するためのリソースがいくつか定義されています。このようなシステム定義のリソースはドキュメント類では使用禁止ということになっていますが，使用したからといって不都合は起こりそうにありません。有用そうなものはどんどん利用してかまわないと思います。それよりも，ユーザーが勝手にリソースをいくつも定義してしまうと可搬性の点で問題が出てくるので，

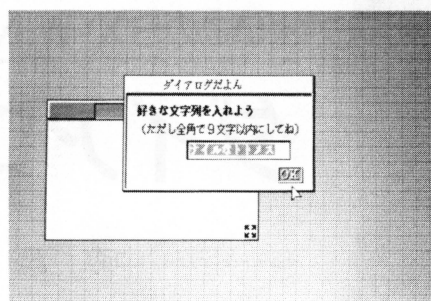
こちらのほうはあまり勧められません。

図１はシステムに初めから定義されているダイアログウィンドウ用のリソースです（本誌５月号のおまけディスク“黄金週間PRO-68K”で拡張されたはずのＳＸ信州用のリソースもまじっています)。文字は書き込まれていませんが，どこかで見たようなウィンドウばかりですね。これらのリソースをユーザーのプログラムで使うことができるように，ウィンドウが表示される位置と制御ボタンの中心位置（どちらもグローバル座標）を図２に示しておきます[3]。

それでは，リソースを利用してダイアログ機能を実現する方法を具体的に説明しましょう。この場合はDMOpenという関数の代わりに，

DMRefer

リスト２の実行結果

という関数でウィンドウをオープンします。DMRefer関数への第１引数がリソースのＩＤ番号です。第２引数はダイアログウィンドウの実体（構造体）を格納する領域へのポインタですが，これに０を指定するとメモリマンを使って領域が確保されます。これはDMOpen関数への第１引数と同じ

リスト３の実行結果

です。つまり，それが０か０以外かでダイアログウィンドウをクローズするときの関数を，

DMDispose

にするか，

DMClose

にするかを切り替えなくてはなりません。

DMRefer関数の第３引数はウィンドウを何番目の位置に表示するかを指定するものですが，－１（一番手前に表示する）以外を指定することはまずないでしょう。

結局，DMRefer関数はウィンドウ情報を引数ではなくリソースから持ってくるほかはDMOpen関数との違いはありません。アイテムリストが不要なため，領域の確保（MMChHdlNew関数)，アイテムリストのコピー(memcpy関数）という手間が省略できるのが魅力です。そのかわり，ウィンドウをオープンしたあとは，ウィンドウ内に文字列を書き込むという処理が必要になります。そして，その後の操作は前回説明した通常のダイアログウィンドウの場合と同一です。前回の復習も兼ねて，リソースを使用しない場合と使用する場合でのダイアログの基本操作の手順を図３にまとめておきます。

リスト２にシステムのリソースを利用するダイアログの例を示します。これはリソースIDが－4114のダイアログで，SX-WINDOWではファイルを指定するときワイルドカードを設定するために使われているものです。これを利用して新しいダイアログウィンドウを作ってみました。文字の表現が自由にできることを示すため，リスト２ではウィンドウ内の一部の文字の大きさや字体を変えたりしています。DMOpen関数を使用する場合は文字の大きさや字体が一意に固定されていたことを考えると，より柔軟なダイアログになっていますね[4]。

リスト３は黄金週間PRO-68Kで配布された「SX信州」で使用されているダイアログのリソース（IDは200）を使用するプログラム例です。このリソースはハイスコア時の名前入力用ですが，文字列（全角９文字

**図１　システムで用意されているダイアログリソース**

1）ID＝－4096

2）ID＝－4097

3）ID＝－4098

4）ID＝－4101

5）ID＝－4112

6）ID＝－4113

7）ID＝－4114

8）ID＝－4115

9）ID＝－4116

10）ID＝200

**図２　ダイアログリソースの表示位置とボタンの位置（座標はだいたいの目安です）**

| ID | ウインドウの位置 | 標準ボタンの位置 | | 用途 |
|---|---|---|---|---|
| －4096 | (270,100)－(500,220) | 確認 | (470,210) | 汎用 |
| －4097 | (256,120)－(490,210) | 設定<br>取消 | (420,200)<br>(460,200) | 画面の記憶用 |
| －4098 | (270, 48)－(500,150) | 中止 | (470,130) | ファイル検索用 |
| －4101 | (270, 48)－(500,180) | ディスクコピー<br>コピーオール<br>取消 | (320,160)<br>(410,160)<br>(470,160) | ディスクコピー用 |
| －4112 | (270,100)－(500,256) | 確認 | (470,240) | シェル情報用 |
| －4113 | (256,120)－(490,210) | 現在の…<br>設定<br>取消 | (330,200)<br>(420,200)<br>(460,200) | スタート画面設定用 |
| －4114 | (290,100)－(480,226) | 設定<br>取消 | (410,210)<br>(450,210) | ワイルドカード用 |
| －4115 | (270, 48)－(500,204) | 中止 | (470,185) | フォーマット用 |
| －4116 | (267,168)－(500,340) | 大文字化<br>小文字化<br>実行<br>取消 | (310,280)<br>(310,330)<br>(430,330)<br>(470,330) | 名前変更用 |
| 200 | (270,100)－(500,224) | OK | (470,210) | ＳＸ信州用 |

以内に限られるけど）を入力するための汎用ダイアログとしても使用することができるのです。

DMReferでリソースIDが200のウィンドウをオープンするとウィンドウ内に文字列を入力するための長方形の領域と標準ボタンが表示されます。ただし，そのままでは文字列を入力することができないので注意しましょう。リソースＩＤが200のダイアログではダイアログアイテムである編集可能文字列の最大文字数が０文字になっているようです。この最大文字数を設定し直さないと文字列を入力することはできません。このためには，dialogという構造体のdTextフィールドのハンドルで示されるテキスト（tEdit）レコードを直接操作して最大文字数（半角文字で考える）を設定しなければなりません。リスト３では，

(*(dialogPtr->dText))->lenMax=18;

の部分がそれに当たります。これは，入力できる全角文字の最大数を９文字（半角文字で18文字）にする記述です。

あと，本質的なことではありませんが，リスト３ではダイアログウィンドウの見栄えをよくするために，グラフマンの，

GMShadowRect

という関数を用いて文字列領域の外枠を影付きの枠線で描き直しています（実際の枠より４ドット外に描いている）。リスト３で，

#define SHADOWRECT

の１行を削除すると，外枠の描き直しを行いませんから，印象を比べてください。ちょっとした工夫で随分雰囲気の異なるダイアログになるものですね。

いい忘れましたが，リソースＩＤが200のダイアログウィンドウは黄金週間PRO-68Kに付属したプログラムによってリソースファイル（SYSTEM.LB）を拡張していないと利用することはできません。このリソースを利用したい人は，付録ディスク３の，

SX/resource/

というディレクトリの，

add_reso.bat

というバッチファイルを実行しておきましょう。

## すべて自前のダイアログ

ダイアログマンを利用すればダイアログ機能を簡単に実現することができますが，それには限界もあります。第１の問題点は，

**ダイアログ実行中は別の処理が停止する**

ということです。つまり，DMControlで（帰還属性を持った）ボタンが押されるまでの間，アプリケーションは制御が返ってくるのをじっと待っているしかありません。ダイアログ実行中にちょっとした別の処理をすることができません。たとえば，ウィンドウに数字を表示しながらカウントダウンを行い，一定時間経ったあとはウィンドウをクローズするという処理はダイアログマンでは不可能です。

また，ダイアログマンではダイアログアイテムとして，

**標準ボタン，セレクトボタン，オルタネートボタンと文字列以外の制御ボタン**

を扱うのは結構面倒臭そうです（少なくとも私にはよくわからない）。スライドボリュームやアップダウンボタンを使いたい場合は困ってしまいます。そこで，これらの問題を解決するためには，

**ダイアログマンを使用しない**

という結論に帰着します（なんと！）。すなわち，ダイアログウィンドウのオープン，ダイアログアイテムの配置や制御をダイアログマンを使用せず，ウィンドウマンやコントロールマンなどのマネージャを個別に使用することでダイアログ機能を実現することになります。これは通常のアプリケーションのウィンドウをダイアログとして利用することにほかなりません。このようなウィンドウでは当然SX-WINDOWのすべての機能を利用することができますし，どのような飾りも思いのままです。ただし，プログラミングが大変になることは言うまでもありません。

リスト４はダイアログマンを使用しないダイアログのプログラム例です。これは現在の時刻を表示しながら，３つの制御ボタン（スライドボリューム）によって値を入力するというダイアログです。ダイアログの終了を待ちながら，同時に刻々と変化する時刻を表示することが特徴です。このような芸当はダイアログマンによるダイアログでは不可能でしょう。リスト４では制御

**図３　ダイアログの基本操作**

方法その１
領域を確保　MMChHdlNew
アイテムリストをコピー　memcpy
ダイアログウィンドウをオープン　DMOpen
方法その２
リソースを使ってダイアログウィンドウをオープン　DMRefer
ダイアログ ウィンドウに文字を書く　GMSetGraph　GMDrawstrZ　etc.
初期値設定不要
アイテムの初期値を設定　DIGet　DITGet　CMValueSet　DITSet etc.
アイテムの描き直し　DMDraw
ボタンが押されるのを待つ　DMControl
アイテムの値の変更等　DIGet　CMValueGet　CMValueSet　DITGet　DITset　etc.
終了ボタンが押された
No
Yes
ダイアログウィンドウをクローズ　DMDisposa　または　DMClose
領域を解放する　MMHdlDispose
終わり

3)　図１や図２ではSYSTEM.LBというファイルの中に定義されているリソースのみを載せてある。ダイアログのリソースはCTRLPNL.LBの中でも定義されている。

4)　もちろん，DMOpen関数でダイアログウィンドウをオープンする場合でも，文字列を後から重ね書きすれば大きさや字体の異なる文字列をウィンドウ上に表示できる。しかし，文字列を２回（ダイアログアイテム内とウィンドウ表示後の描画）指定しなければならないのは二度手間である。

リスト4の実行結果

ボタンを持った通常のウィンドウのプログラムと同じく，

WMOpen

関数でウィンドウをオープンし，

CMOpen

関数で制御ボタン（この場合はスライドボリューム）を用意し，その後にイベント待ちの無限ループを作ることでユーザーからの入力（これがダイアログ機能になる）に応対しています。このとき，イベント待ちのループを1周回るたびにウィンドウに表示されている時刻を描き換えるようにしてダイアログ機能との同時動作を行っています。

さて，リスト4のプログラムが通常のウィンドウのプログラムと大きく異なるのは，発生したイベントを，

TSEventAvail

関数ではなく，

EMGet

関数によって参照している点です。その機能的な違いは，TSEventAvail関数がタスクの切り替えを行うのに対して，EMGet関数はタスクの切り替えを行わないことにあります。

タスクの切り替えが発生すると，(マウスで触ることによって）別のウィンドウがアクティブになります（すなわち処理がそちらに移る）から，これは，

**別のウィンドウの処理を停止する**

というダイアログの精神に反することになります[5]。

一方，EMGet関数ではタスクが切り替わりませんからほかのウィンドウに制御が移るということはありません。このとき，ダイアログウィンドウ以外に対してイベントが発生するとは考えられませんから，私たちはマウスボタンダウンイベントやキーダウンイベントが表示されているダイアログウィンドウに対して発生したものと決めつけてその後の処理を行ってかまいません。リスト4でイベントが自分のウィンドウに対して発生したのかどうか調べることをしていないのはそのためです。この点，通常のウィンドウのプログラムよりも処理が簡単になります。

ところで，EMGet関数ではアイドルイベントを取り出すことはできないようです（というより，EMGet関数使用中はアイドルイベントが発生しない？）。このため，EMGet関数を使ったダイアログでは通常のウィンドウのプログラムで行っていたような，アイドルイベント時に何かの処理を行うということはできません。そこで，リスト4でも（通常ならアイドルイベント発生時に行われるはずの）時刻表示の処理をEMGet関数を呼ぶ直前に行っています。

あとリスト4では，こまめに字体を変えたり，ダイアログウィンドウの外をマウスでクリックしたらビープ音を鳴らす（DMBeep関数を使用）などの処理などを行っていますがこれらは枝葉末節の処理です[6]。処理の大まかな流れはわかりますね。また，リスト4ではリスト3でも使用した，

GMShadowRect

関数を多用しています。この関数を使用するとダイアログの見た目がとてもカッコよくなりますね（くせになりそうです）。

---

5) タスク切り替えが発生することを積極的に利用するダイアログも考えることができるが，ここでは考えない。

6) リスト4でCMOpen後のエラーチェックを行ってないのは手抜きである。これらは枝葉末節とはいわないが，この処理のためにリストの行数が増えて読みにくくなるのが嫌だっただけである。

＊

前回と今回の2回にわたりSX-WINDOWのダイアログ機能について紹介してきましたが，理解できたでしょうか。DMErrorやDMBeepといった使用頻度の高そうな関数の説明を省いてありますが，これらの操作については迷うことはないでしょう。この2回でダイアログ作りの基本は一応網羅したつもりですから，それらをうまく応用して，皆さんも使いやすいダイアログウィンドウを作ってみましょう。

次回の予定は未定です。何が飛び出すかお楽しみに（……といいつつ何にしようか考えている)。

### リスト1　2つの編集可能テキストを持つダイアログ

```
 1: /***************************************
 2:  *  ダイアログのサンプルプログラム  *
 3:  *                                 *
 4:  *          1991.4.6    中森  章   *
 5:  ***************************************/
 6: #include <stdio.h>
 7: #define __POINT_T        /* point_t 型を使う */
 8: #include <sxlib.h>
 9: #define FALSE   0
10: #define TRUE    ~FALSE
11: /*
12:     ここでダイアログウィンドウに関する定数を設定
13: */
14: #define DWINDEFID        (WI_STD<<4)
15: #define DWINTITLE        "¥020ダイアログだよん"
16: /*
17:     アイテムリスト（sxlib.h内の定義だけで書くのはキツイ）
18: */
19: typedef struct dlgItem2 {
20:     long            dlgIHdl;
21:     rect            dlgIBounds;
22:     unsigned char dlgIType;
23:     unsigned char dlgISize;
24:     unsigned char dlgIData[32]; /* 初期値データサイズを */
25:                                 /* 32バイトに固定した型 */
26: } dlgItem2;
27:
28: struct {
29:     short    itemNo;
30:     dlgItem2 dItem1;
31:     dlgItem2 dItem2;
32:     dlgItem2 dItem3;
33:     dlgItem2 dItem4;
34:     dlgItem2 dItem5;
35:     dlgItem2 dItem6;
36: } dItemList ={
37:         6-1,                    /* ダイアログの個数－1 */
38:     {
39:         0,                      /* ハンドル等         */
40:         {256-8-46-42,128-8-18,256-8-46,128-8},  /* 境界 */
41:         DT_STDBTN,              /* 標準ボタン         */
42:         32,                     /* 初期値データサイズ！ */
43:         "¥007 Ｏ Ｋ "           /* 初期値データ       */
44:     },
45:     {
46:         0,                      /* ハンドル等         */
47:         {256-8-42,128-8-18,256-8,128-8},        /* 境界 */
48:         DT_STDBTN,              /* 標準ボタン         */
49:         32,                     /* 初期値データサイズ！ */
50:         "¥007 取 消 "           /* 初期値データ       */
51:     },
52:     {
53:         0,                      /* ハンドル等         */
54:         {16,16,16+48,28},       /* 境界               */
55:         DT_STCTXT+DT_DISABL,    /* 固定ﾃｷｽﾄ+未帰還属性 */
56:         32,                     /* 初期値データサイズ  */
57:         "¥000¥010文字列1"       /* 初期値データ,左寄せ */
58:     },
59:     {
60:         0,                      /* ハンドル等         */
61:         {16,40,16+48,52},       /* 境界               */
62:         DT_STCTXT+DT_DISABL,    /* 固定ﾃｷｽﾄ+未帰還属性 */
63:         32,                     /* 初期値データサイズ  */
64:         "¥000¥010文字列2"       /* 初期値データ,左寄せ */
65:     },
66:     {
67:         0,                      /* ハンドル等         */
68:         {68,16,68+128,28},      /* 境界 1文字くらい余分
                                       に領域をとっておく */
69:         DT_EDTTXT+DT_DISABL,    /* 編集可能テキスト   */
70:         32,                     /* 初期値データサイズ  */
71:         "¥024びびるまびびるま"  /* 初期値ﾃﾞｰﾀ 最大20文字*/
72:     },
73:     {
74:         0,                      /* ハンドル等         */
75:         {68,40,68+128,52},      /* 境界 1文字くらい余分
                                       に領域をとっておく */
76:         DT_EDTTXT+DT_DISABL,    /* 編集可能テキスト */
```

```
 77:         32,                      /* 初期値データサイズ   */
 78:         "¥024びんぶるばんぶる"  /* 初期値ﾃﾞｰﾀ 最大20文字*/
 79:    }
 80: };
 81:
 82: /*
 83:    ダイアログを開く位置（中央よりも少し上にしてある）
 84: */
 85: rect dlBounds={ 384-128,256-64-20,384+128,256+64-20 };
 86:
 87:
 88: /*
 89:    ダイアログ内のアイテムの値（初期値）
 90: */
 91: char    dI5Value[21]="¥020びびるまびびるま";
 92: char    dI6Value[21]="¥020びんぶるばんぶる";
 93: char    t5Value[21];              /* アイテム5の退避用 */
 94: char    t6Value[21];              /* アイテム6の退避用 */
 95:
 96: int     curText;
 97: /*
 98:    フィルター関数
 99: */
100: MyFilter(Dialog,ev)
101: dialog *Dialog;
102: event  *ev;
103: {
104:    point_t okbtn;
105:    int     x,y;
106:
107:    if( ev->eWhat == E_KEYDOWN ){
108:       switch((short)(ev->eWhom)){
109:       case 9:
110:          ev->eWhom &= 0xffff0000;
111:          ev->eWhom |= 32;
112:          break;
113:       case 13:
114:          okbtn.p.x=384+128-70;   /* OKボタン */
115:          okbtn.p.y=256+64-20-10;
116:          ev->eWhere=okbtn;
117:          ev->eWhat =E_MSLDOWN;
118:          break;
119:       }
120:    }
121:    else if(ev->eWhat == E_MSLDOWN ){
122:       okbtn=ev->eWhere;
123:       x = (ev->eWhere).p.x - dlBounds.left;
124:       y = (ev->eWhere).p.y - dlBounds.top;
125:       if( x>=68 && x<=(68+128)){
126:          if( y>=16 && y<=28 ){ /* ｱｲﾃﾑ5が選択された */
127:             t6Value[0]=TMGetText(Dialog->dText,t6Value+1,20);
128:             curText=5;   /* 編集テキストの切り替え */
129:          }
130:          else if ( y>=40 && y<= 52){ /* ｱｲﾃﾑ6が選択された */
131:             t5Value[0]=TMGetText(Dialog->dText,t5Value+1,20);
132:             curText=6;
133:          }
134:       }
135:    }
136:    return 0;
137: }
138:
139:
140: /*********************************************
141:       ダイアログを使う
142: *********************************************/
143: doDialog()
144: {
145:     dialog  *dialogPtr;
146:     dlgIList **dIHdl;
147:     int      ditem;
148:
149:     short    DI_T;
150:     int      DI_H;
151:     rect     DI_R;
152:
153: /* (1) 領域を確保する */
154:
155:     dIHdl=(dialog**)MMChHdlNew( sizeof(dItemList) );
156:     if( dIHdl == NULL ){
157:         DMError(0x101,"領域確保に失敗しました。");
158:         return ( FALSE );
159:     }
160:
161: /* (2) アイテムリストをコピーする */
162:
163:     memcpy(*dIHdl,&dItemList,sizeof(dItemList));
164:
165: /* (3) ウインドウをオープンする */
166:
167:     dialogPtr=DMOpen(NULL,&dlBounds,DWINTITLE,TRUE,DWINDEFID,
168:            (window *)-1,FALSE,TSGetID(),dIHdl);
169:     if( dialogPtr == NULL ){
170:         MMHdlDispose(dIHdl);
171:         DMError(0x101,"ウインドウがオープンできません。");
172:         return( FALSE );
173:     }
174:
175: /* (3-1) アイテムの初期値を入れる */
176:
177:     DIGet(dialogPtr,5,&DI_T,&DI_H,&DI_R);
178:     DITSet(DI_T,DI_H,dI5Value);
179:     memcpy(t5Value,dI5Value,21);
180:
181:     DIGet(dialogPtr,6,&DI_T,&DI_H,&DI_R);
182:     DITSet(DI_T,DI_H,dI6Value);
183:     memcpy(t6Value,dI6Value,21);
184:
185:     DMDraw(dialogPtr);   /* 描き直し */
186:
187:     curText=5;       /* 現在の編集テキストのアイテム番号 */
188:
189: /* (4) ボタンが押されるのを待つ */
190:
191:     while(1){
192:         ditem=DMControl((void*)MyFilter );
193:
194: /* (5) 選択されたアイテムに応じた処理をする */
195:
196:         if(ditem==1 || ditem==2) break; /* OKか取消か */
197:     }
198:
199: /* (5-1) OKならアイテムの値を更新する */
200:
201:     if(ditem==1){
202:         if(curText==5){
203:             DIGet(dialogPtr,5,&DI_T,&DI_H,&DI_R);
204:             DITGet(DI_T,DI_H,dI5Value);
205:             memcpy(dI6Value,t6Value,21);
206:         }
207:         else{
208:             DIGet(dialogPtr,6,&DI_T,&DI_H,&DI_R);
209:             DITGet(DI_T,DI_H,dI6Value);
210:             memcpy(dI5Value,t5Value,21);
211:         }
212:     }
213:
214: /* (6) ウインドウをクローズする */
215:
216:     DMDispose(dialogPtr);
217:
218: /* (7) 領域を解放する */
219:
220:     MMHdlDispose(dIHdl);
221:
222: };
```

リスト2　リソースを利用するダイアログ（その1）

```
 1: /**************************************
 2:  *  ダイアログのサンプルプログラム  *
 3:  *                                    *
 4:  *   DMRefer を DLOG=-4114 で使用    *
 5:  *                                    *
 6:  *            1991.5.12    中森 章  *
 7:  **************************************/
 8: #include <stdio.h>
 9: #define __POINT_T         /* point_t 型を使う */
10: #include <sxlib.h>
11: #define FALSE   0
12: #define TRUE    ~FALSE
13: /*
14:    フィルター関数
15: */
16: MyFilter(Dialog,ev)
17: dialog *Dialog;
18: event  *ev;
19: {
20:     point_t okbtn;
21:
22:     if( ev->eWhat == E_KEYDOWN ){
23:         if((short)(ev->eWhom)==13){
24:             okbtn.p.x=400;
25:             okbtn.p.y=210;
26:             ev->eWhere=okbtn;
27:             ev->eWhat =E_MSLDOWN;
28:         }
29:     }
30:     return 0;
31: }
32:
33:
34: /*********************************************
35:       ダイアログを使う
36: *********************************************/
37: doDialog()
38: {
39:     dialog   *dialogPtr;
40:     point_t  pt;
41:     int      ditem;
42:
43:     short    DI_T;
44:     int      DI_H;
45:     rect     DI_R;
46:
47:     int      tmp1;
48:     int      tmp2;
49:
50:     static   chk1=0;
51:     static   chk2=0;
52:
53: /* (1) ウィンドウをオープンする */
54:
55:     dialogPtr=DMRefer(-4114,NULL,(window*)-1);
56:     if( dialogPtr == NULL ){
57:         DMError(0x101,"ウインドウがオープンできません。");
58:         return( FALSE );
59:     }
60:
61: /* (2) ウィンドウに飾りをつける（文字を書く） */
62:
63:     GMSetGraph(dialogPtr);  /* current graph を設定 */
64:
65:     pt.p.x=48;
```

```
 66:     pt.p.y=4;
 67:     GMShadowStrZ("ダイアログだよん",pt);
 68:
 69:     GMFontFace(G_BOLD);       /* ボールド体 */
 70:     GMFontKind(G_ROM16);      /* 16×16ドットフォント */
 71:     pt.p.x=10;
 72:     pt.p.y=36;
 73:     GMMove(pt);
 74:     GMDrawStrZ("環境設定しましょうね");
 75:
 76:     GMFontFace(0);            /* 字体を初期化 */
 77:     GMFontKind(G_ROM12);      /* 12×12ドットフォント */
 78:     pt.p.x=48;
 79:     pt.p.y=68;
 80:     GMMove(pt);
 81:     GMDrawStrZ("33MHzで実行"); /* 冗談です */
 82:
 83:     pt.p.x=48;
 84:     pt.p.y=84;
 85:     GMMove(pt);
 86:     GMDrawStrZ("68882を使用"); /* 冗談ですよ */
 87:
 88:
 89: /* (3) 制御ボタンの初期値を設定する */
 90:
 91:     DIGet(dialogPtr,3,&DI_T,&DI_H,&DI_R);
 92:     tmp1=chk1;
 93:     CMValueSet(DI_H,tmp1);
 94:
 95:     DIGet(dialogPtr,4,&DI_T,&DI_H,&DI_R);
 96:     tmp2=chk2;
 97:     CMValueSet(DI_H,tmp2);
 98:
 99: /* (4) ボタンが押されるのを待つ */
100:
101:     while(1) {
102:
103:         ditem=DMControl((void*)MyFilter );
104:
105: /* (5) 選択されたアイテムに応じた処理をする */
106:
107:         DIGet(dialogPtr,ditem,&DI_T,&DI_H,&DI_R);/* ハンドル
を得る */
108:
109:         switch(ditem){
110:         case 1:
111:          chk1=tmp1;                /* 値をセーブ */
112:          chk2=tmp2;
113:          break;
114:         case 3:
115:          tmp1=CMValueGet(DI_H);    /* 値を得る */
116:          tmp1 ^= 1;                /* 値を反転 */
117:          CMValueSet(DI_H,tmp1);    /* 値を更新 */
118:          CMDrawOne(DI_H);          /* 描き直す */
119:          break;
120:         case 4:
121:          tmp2=CMValueGet(DI_H);    /* 値を得る */
122:          tmp2 ^= 1;                /* 値を反転 */
123:          CMValueSet(DI_H,tmp2);    /* 値を更新 */
124:          CMDrawOne(DI_H);          /* 描き直す */
125:          break;
126:         }
127:         if(ditem==1 || ditem==2) break;
128:     }
129:
130:
131: /* (6) ウインドウをクローズする */
132:
133:     DMDispose(dialogPtr);
134: }
```

## リスト3 リソースを利用するダイアログ（その2）

```
  1: /**********************************************
  2:  *   ダイアログのサンプルプログラム             *
  3:  *                                              *
  4:  *    DMRefer を DLOG=200（SX信州用）で使用     *
  5:  *                                              *
  6:  *                     1991.5.12    中森 章     *
  7:  **********************************************/
  8: #include <stdio.h>
  9: #define __POINT_T         /* point_t 型を使う */
 10: #include <sxlib.h>
 11: #define FALSE   0
 12: #define TRUE    ~FALSE
 13: /*
 14:       フィルター関数
 15: */
 16: MyFilter(Dialog,ev)
 17: dialog *Dialog;
 18: event  *ev;
 19: {
 20:     point_t okbtn;
 21:
 22:     if( ev->eWhat == E_KEYDOWN ){
 23:         if((short)(ev->eWhom)==13){
 24:             okbtn.p.x=470;
 25:             okbtn.p.y=200;
 26:             ev->eWhere=okbtn;
 27:             ev->eWhat =E_MSLDOWN;
 28:         }
 29:     }
 30:     return 0;
 31: }
 32:
 33: #define SHADOWRECT        /* 文字列の枠に影をつける */
 34: /*********************************************
 35:       ダイアログを使う
 36: *********************************************/
 37: char *
 38: doDialog()
 39: {
 40:     dialog   *dialogPtr;
 41:     point_t  pt;
 42:     int      ditem;
 43:
 44: #ifdef SHADOWRECT
 45:     rect     strRect;
 46: #endif
 47:
 48:     short    DI_T;
 49:     int      DI_H;
 50:     rect     DI_R;
 51:
 52:     static   char string[20]="¥020ナイルなトトメス";
 53:
 54: /* (1) ウィンドウをオープンする */
 55:
 56:     dialogPtr=DMRefer(200,(dialog*)NULL,(window*)-1);
 57:     if( dialogPtr == NULL ){
 58:         DMError(0x101,"ウインドウがオープンできません。");
 59:         return( FALSE );
 60:     }
 61:
 62: /* (2) ウィンドウに飾りをつける（文字を書く） */
 63:
 64:     GMSetGraph(dialogPtr);  /* current graph を設定 */
 65:
 66:     pt.p.x=48;
 67:     pt.p.y=4;
 68:     GMShadowStrZ("ダイアログだよん",pt);
 69:
 70:     GMFontFace(G_BOLD);       /* ボールド体 */
 71:     pt.p.x=16;
 72:     pt.p.y=32;
 73:     GMMove(pt);
 74:     GMDrawStrZ("好きな文字列を入れよう");
 75:
 76:     GMFontFace(0);                   /* 字体を初期化する */
 77:     pt.p.x=16;
 78:     pt.p.y=52;
 79:     GMMove(pt);
 80:     GMDrawStrZ("（ただし全角で9文字以内にしてね）");
 81:
 82:     GMFontFace(G_BOLD);       /* ボールド体 */
 83:
 84: #ifdef SHADOWRECT
 85:     strRect.left   =80-4;
 86:     strRect.top    =74-4;
 87:     strRect.right  =200+4;
 88:     strRect.bottom=88+4;
 89:
 90:     GMShadowRect(&strRect);
 91: #endif
 92:
 93: /* (3) 制御ボタンの初期値を設定する */
 94:
 95:     (*(dialogPtr->dText))->lenMax = 18; /*最大文字数設定*/
 96:
 97:     DIGet(dialogPtr,1,&DI_T,&DI_H,&DI_R); /*アイテム番号は1*/
 98:     DITSet(DI_T,DI_H,string);         /* 文字列を設定して */
 99:     DMDraw(dialogPtr);                /* 描き直し */
100:
101: /* (4) ボタンが押されるのを待つ */
102:
103:     ditem=DMControl((void*)MyFilter );
104:
105: /* (5) 選択されたアイテムに応じた処理をする */
106:
107:     DIGet(dialogPtr,1,&DI_T,&DI_H,&DI_R);/* 文字列のハン
                                                ドルを得る */
108:     DITGet(DI_T,DI_H,string);    /* 文字列を得る */
109:
110:     GMFontFace(0);               /* 字体を初期化する */
111:
112: /* (6) ウインドウをクローズする */
113:
114:     DMDispose(dialogPtr);
115:
116:     return (string);
117: }
```

## リスト4 ダイアログマンを使わないダイアログ

```
  1: /**********************************************
  2:  *   ダイアログのサンプルプログラム             *
  3:  *                                              *
  4:  *   すべて自前のダイアログ                     *
  5:  *                                              *
  6:  *                     1991.5.12    中森 章     *
  7:  **********************************************/
  8: #include <stdio.h>
  9: #include <time.h>
 10: #define __POINT_T     /* point_t 型を使う */
 11: #include <sxlib.h>
 12: #ifndef __STDC__
 13: #define time_t long  /* XC Ver.1 ではtime_tの定義がない */
 14: #endif
```

▶ああ，なんてことだ。プレゼントに当たったのはうれしいのだが，そのプレゼントはもうすでに買ってしまったのだ。神様のばかやろー！　　毛塚　健次(18)栃木県

```
15: #define FALSE   0
16: #define TRUE    ~FALSE
17: /*
18:       サブルーチン群
19: */
20: static
21: ShadowBox(L,T,R,B)
22: int  L,T,R,B;
23: {
24:     rect r;
25:     r.left   = L;
26:     r.top    = T;
27:     r.right  = R;
28:     r.bottom = B;
29:     GMShadowRect(&r);
30: }
31:
32: static
33: control **
34: CtrlOpen(L,T,R,B,ID,V,TITL,P)
35: int     L,T,R,B,ID,V;
36: char    *TITL;
37: dialog *P;
38: {
39:     control **c;
40:     rect    r;
41:     r.left   = L;
42:     r.top    = T;
43:     r.right  = R;
44:     r.bottom = B;
45:     c=CMOpen(P,&r,TITL,TRUE,V,0,100,(ID<<4),0);
46:     return (c);
47: }
48:
49: static
50: DrawPercent(L,T,R,B,V)
51: int  L,T,R,B,V;
52: {
53:     point_t pt;
54:     char    BUF[10];
55:
56:     ShadowBox(L,T,R,B);
57:     sprintf(BUF,"%3d",V);
58:     pt.p.x=L;
59:     pt.p.y=T+2;
60:     GMMove(pt); GMDrawStrZ(BUF);
61: }
62:
63: static
64: DrawTime()
65: {
66:     point_t         pt;
67:     static time_t tp0=0;
68:     time_t    tp;
69:     char      BUF[26];
70:
71:     tp=time(NULL);  /* 時刻を得る */
72:     if(tp0==tp)     /* 前の時刻と同じなら何もしない */
73:         return;
74:     tp0=tp;
75:     strcpy(BUF,ctime(&tp)+11);  /* 時刻を文字に変換 */
76:     BUF[8]=NULL;           /* 不要部分を消す */
77:     ShadowBox(134,26,192,44);
78:     pt.p.x=138;
79:     pt.p.y=28;
80:     GMMove(pt); GMDrawStrZ(BUF);/* 時刻を書く */
81: }
82:
83: /*********************************************
84:       ダイアログを使う
85: *********************************************/
86: doDialog()
87: {
88:     window  *dialogPtr;
89:     rect    winRect;
90:     event   eventRec;
91:
92:     control **selHdl;
93:     control **okHdl;
94:     control **canHdl;
95:
96:     control **ctrl1Hdl;     /* コントロールへのハンドル */
97:     static  ctrl1Val=0;     /* コントロールの値         */
98:
99:     control **ctrl2Hdl;     /* コントロールへのハンドル */
100:    static  ctrl2Val=0;     /* コントロールの値         */
101:
102:    control **ctrl3Hdl;     /* コントロールへのハンドル */
103:    static  ctrl3Val=0;     /* コントロールの値         */
104:
105:    point_t pt;
106:    int     endDLOG;
107:    int     part;
108:
109: /* (1) ウィンドウをオープンする */
110:
111:    winRect.left   = 260;
112:    winRect.top    = 100;
113:    winRect.right  = 484;
114:    winRect.bottom = 260;
115:
116:    dialogPtr=WMOpen(NULL,&winRect,"¥012ダイアログ",TRUE,(38<<4),
117:                     (window *)-1,TRUE,TSGetID());
118:
119: /* (2) ウィンドウに飾りをつける(文字を書く) */
120:
121: /* (3) 制御ボタンの初期値を設定する */
122:
123:    GMSetGraph(dialogPtr);  /* current graph を設定 */
124:
125:    DrawTime();
126:
127:    GMFontFace(G_OLINE);     /* 中抜き */
128:
129:    pt.p.x=48;
130:    pt.p.y=4;
131:    GMMove(pt); GMDrawStrZ("平成3年度予算(案)");
132:
133:    GMFontFace(G_BOLD);      /* ボールド体 */
134:
135:    pt.p.x=16;
136:    pt.p.y=50;
137:    GMMove(pt); GMDrawStrZ("今年度の予算を決定してください");
138:
139:    pt.p.x=16;
140:    pt.p.y=72;
141:    GMMove(pt); GMDrawStrZ("交通局");
142:    pt.p.x=202;
143:    GMMove(pt); GMDrawStrZ("%");
144:    ctrl1Hdl=CtrlOpen(68,72,168,88,CI_SLDVOL,ctrl1Val,"",dialogP
tr);
145:    DrawPercent(176,72,198,88,ctrl1Val);
146:
147:    pt.p.x=16;
148:    pt.p.y=92;
149:    GMMove(pt); GMDrawStrZ("警察署");
150:    pt.p.x=202;
151:    GMMove(pt); GMDrawStrZ("%");
152:    ctrl2Hdl=CtrlOpen(68,92,168,108,CI_SLDVOL,ctrl2Val,"",dialog
Ptr);
153:    DrawPercent(176,92,198,108,ctrl2Val);
154:
155:    pt.p.x=16;
156:    pt.p.y=112;
157:    GMMove(pt); GMDrawStrZ("消防署");
158:    pt.p.x=202;
159:    GMMove(pt); GMDrawStrZ("%");
160:    ctrl3Hdl=CtrlOpen(68,112,168,128,CI_SLDVOL,ctrl3Val,"",dialo
gPtr);
161:    DrawPercent(176,112,198,128,ctrl3Val);
162:
163:    okHdl =CtrlOpen(146,134,180,154,CI_STDBTN,0,"¥005決 定",dial
ogPtr);
164:
165:    canHdl=CtrlOpen(184,134,218,154,CI_STDBTN,0,"¥005取 消",dial
ogPtr);
166:
167:    CMDraw(dialogPtr);
168:
169: /* (4) ボタンが押されるのを待つ */
170:
171: /* (5) 選択されたアイテムに応じた処理をする */
172:
173:    endDLOG=FALSE;
174:    while( !endDLOG ){
175:        GMFontFace(0);                  /* 字体初期化 */
176:        DrawTime();                     /* 定期的に時刻を書く */
177:        GMFontFace(G_BOLD);             /* ボールド体 */
178:        EMGet(EM_EVERY,&eventRec);/* イベントを見る */
179:        switch( eventRec.eWhat ){
180:        case E_MSLDOWN:
181:            pt=eventRec.eWhere;      /* マウスの座標 */
182:            if(pt.p.x<winRect.left || pt.p.x>winRect.right
183:            || pt.p.y<winRect.top  || pt.p.y>winRect.bottom){
184:                DMBeep(2);       /* ダイアログの外側 */
185:                break;
186:            }
187:            part=SXCallCtrlM(dialogPtr,&eventRec,
188:                NULL,NULL,NULL,&selHdl);
189:            if(selHdl==okHdl){
190:                endDLOG=TRUE;
191:                ctrl1Val=CMValueGet(ctrl1Hdl);
192:                ctrl2Val=CMValueGet(ctrl2Hdl);
193:                ctrl3Val=CMValueGet(ctrl3Hdl);
194:            }
195:            else if(selHdl==canHdl){
196:                endDLOG=TRUE;
197:            }
198:            else if(selHdl==ctrl1Hdl){
199:                DrawPercent(176,72,198,88,CMValueGet(selHdl));
200:            }
201:            else if(selHdl==ctrl2Hdl){
202:                DrawPercent(176,92,198,108,CMValueGet(selHdl));
203:            }
204:            else if(selHdl==ctrl3Hdl){
205:                DrawPercent(176,112,198,128,CMValueGet(selHdl));
206:            }
207:            break;
208:        case E_KEYDOWN:
209:            if((short)(eventRec.eWhom)!=13)
210:                break;
211:            endDLOG=TRUE;
212:            CMShine(okHdl,C_INBTTN);
213:            ctrl1Val=CMValueGet(ctrl1Hdl);
214:            ctrl2Val=CMValueGet(ctrl2Hdl);
215:            ctrl3Val=CMValueGet(ctrl3Hdl);
216:            break;
217:        }
218:    }
219:
220: /* (6) ウインドウをクローズする */
221:
222:    GMFontFace(0);  /* 字体初期化 */
223:
224:    CMDispose(okHdl);
225:    CMDispose(canHdl);
226:    CMDispose(ctrl1Hdl);
227:    CMDispose(ctrl2Hdl);
228:    CMDispose(ctrl3Hdl);
229:    WMDispose(dialogPtr);
230:
231: }
```

▶遥かなるオーガスタの女性キャディはSHIFTキーを押して起動すると現れますよ。美女が4人，どの娘にするかお楽しみ。　　加藤　晴三(36)福岡県

ようこそここへC言語

# [第9回] 式と演算子って何だろう

Nakamori Akira
中森 章

プログラムとは切っても切れない関係にある式と演算子。これらはおおむねどのプログラミング言語でも共通ですが，よりいっそう理解を深めるためにも，今回はC言語における独特の式と演算子の使い方を解説します。

X68000用の「パロディウスだ！」では無限にコンティニューができるおかげで，シューティングゲームの苦手な私でも晴れてエンディングを見ることができました[0]。2周目は1面クリアがやっとですが，なぜか感無量の中森章です。

さて，今回は式と演算子について解説したいと思います。これまでの連載でも式や演算子は当たり前のものとして説明なしに使用してきました。実際，C言語で使用する式や演算子は，BASICやFORTRANなどのプログラミング言語とほぼ共通です。また，プログラミング言語での式や演算子は数学での記法と違和感なく使用できるようになっています。そのため，特に説明をしなくても常識的な線で使用することができたはずです。

こういうわかりきったことは概略だけを説明することにして，今回はC言語に独特な式や演算子を中心に解説します。これを修得すればあなたのプログラムがよりC言語のプログラムらしくなること請け合いです。

0) スーパーファミコン版の「グラディウスIII」は3回しかコンティニューできない。当然エンディングなんか見たことない。

## 式や演算子というもの

これまでなにげなく使用してきた式と演算子というものについて考えてみましょう。プログラミング言語でいう式とは数学でいう式と同じものです。つまり，数値や変数名を＋や－などの符号（演算子）でつないだものです。たとえば，

**1+2+3-4**

**x+2*y**

が式の例となります(わかりますよね)。では，式とはなんのために存在するのでしょうか。それは数値や変数の値をどのような順序で計算していくかを指示するためのものなのです。そして，式で示される手順で計算した結果をその式の値といいます。たとえば，上の，

1+2+3-4

という式は，

1に2を加え，

その結果に3を加え，

その結果から4を引く

という手順を示したもので，値は2になります[1]。これは，私たちが，

＋は左辺と右辺の和を計算する演算子であり，

－は左辺と右辺の差を計算する演算子であり，

＋や－の計算は左から順に行われていく

という事実を数学（や算数）の常識として与えられているためにわかる手順です。また，

1+2*3+4

という式が与えられた場合は，

2を3倍し，

1とその結果を加え，

その結果と4を加える

という手順を示しています。これは上の常識に加えて，

＊は左辺と右辺の積を計算する演算子であり，

＊は＋よりも先に計算する

という常識によって手順を決定しています（この式の値は11ですね)。

式というものは計算の手順を示すものですが，その具体的な計算の手順は式に使われる演算子によって決定されます。いま，仮に@，#，$という演算子があるとして，

1@2#3$4+5

という式はどういう手順で計算したらよいのでしょう。それは誰にもわかりません。なぜなら，まず第1に，@，#，$といった演算子がどのような演算（計算）を行うのかがわかりません。もし，それらがわかっているとしても，それらの演算子間の優先順位が不明です。優先順位とはいくつかの演算が並んでいる場合に，どの演算から先に行ったらよいかを示す決まりです。たとえば，加算と乗算を示す演算子である＋と＊では，＊のほうが優先順位が高いと決められているので，＋と＊が入った式では＊の演算を先に行います。

このように，式はそれで使用される演算子の演算内容と，その演算子の他の演算子に対する優先順位がわかってないと計算手順を知ることができません（式を計算できない)。

さて，これを逆に考えると，すべての演算子間での優先順位を決定してさえおけば，どのような演算をする演算子があっても，式を計算できることがわかります。

---

1) この式の計算手順は，
　2と3を加え，
　1をその結果に加え，
　その結果から4を引く
というように考えることもできる(実際，計算結果は正しい)。ただし，これは＋や－といった演算子に結合則
　(A＋B)＋C＝A＋(B＋C)
が成立するために成り立つことであり，一般には計算手順を変えると正しい計算結果は得られない。

## C言語で使用する演算子

プログラミングの大部分は式を記述することです。プログラミング言語の式では私たちが日常で使用する加減乗除（＋－＊/）以外の演算子がたくさん登場しますが，演算子の演算内容と演算子間の優先順位さえしっかりと押さえておけば式を理解するのはたやすいことです。表1と表2にC言語で使用する演算子をまとめておきます。表1が式に使われる演算子の演算内容，表2が演算子間の優先順位です。表1や表2の説明文の中にはこれまで説明してない言葉も出てきますが，とりあえずは無視しておいてください。以下ではC言語で使用される式や演算子のうち特徴的なものを順次解説していくことにしましょう。

**●基本式**

基本式はC言語に特徴的というわけではありませんが，大事な概念なので説明しておきましょう。基本式とは読んで字のごとく（演算の）基本となる式のことです。これは式というよりも値そのものといったほうがピンとくるかもしれません。式とはいくつかの値を演算子で関係づけたものですが，演算子の演算対象となる値を式として分類したものが基本式です。これは数値（文字定数を含む）や文字列や変数など，計算をしなくてもそれ自身が値となりうるものを意味します。つまり，

```
'a'
x
123
"eternal wind"
```

などといったものです。これらの基本式については計算しろといわれても，変数のときは値を取り出すことと考えてもいいのですが，それ自身を値とする以外にやりようがありませんね（だって演算子がありませんからね）。

これらはともかく，式を（ ）で囲んだものも基本式になります。

```
(1+2)+(3+4)
```

という式は（1＋2）という基本式と（3＋4）という基本式を＋という演算子で演算することを意味します。（1＋2）のように（ ）がついた基本式の値は（ ）内の式の値となります。したがって，

```
(1+2)+(3+4)
```

という式を計算する（値を求める）ためには（1＋2），（3＋4）といった基本式の値が求まっていることが前提となります。つまり実質的には，（ ）が含まれる式では（ ）内を先に計算しなければなりません。これは基本式という概念を利用して，（ ）をつけることによって演算子の優先順位を最優先に変更できるという数学の常識を実現していることになります。

このようにC言語の式においても（ ）をつけることによってその部分を先に計算することができるようになっています。表1や表2に示すようにC言語には多くの演算子が用意されていますが，それら演算子の暗黙的な優先順位にたよった式の記述は式を見にくくするだけです。優先順位のわかりきった式でない限り，（ ）をつけて計算の順序を明示するのがいいと思います[2)]。

```
ch>='a' && ch<='z'
```

という式を書いたとき，これが本当に，

```
(ch>='a') && (ch<='z')
```

という意味に解釈されるのか不安になって，優先順位の表（表2など）でいちいち確かめるのは時間の無駄ですからね。

**●代入演算子**

式というものは計算結果である値を返す。これは常識ですが，C言語では値を返すよりも副作用が発生することを目的とした演算子があります。それが代入演算子です。この代表例は＝という演算子です。この演算子を用いた，

```
x=123
```

などという式は，式の値を求めるというよりも，＝の右辺の値（123）を左辺で示される変数（x）に代入することを目的としています。＝はその左辺と右辺に基本式を必要とする2項演算子ですが，その演算には右辺の値しか使用されず，副作用として左辺で示す変数に右辺の値を代入します。便宜上＝の左辺は変数としましたが，値を代入できるもの（値を変更できるもの，すなわち値の格納場所があるもの）であればなんでもかまいません。たとえば，

```
p[10]
```

などといった配列の要素でもいいのです。逆に，値を変更できないものを＝の左辺に持ってくるとエラーになります。これが，同じ2項演算子とはいえ，＝が加減乗除（＋－＊/）などの演算子と異なる点なのです。

```
1=x
2*x=12
```

などという式は許されませんが，＝を＋で置き換えた，

```
1+x
2*x+12
```

はどちらも正しい式です。

ところで，BASICやFORTRANなどのプログラミング言語では代入は式でなく文として定義してあります。すなわち，代入文を，

**変数＝式**

などと定義し，＝を演算子ではなく「代入文」という文を示す目印として用いているのです。この代入文の意味は，もちろん，右辺の式の値を左辺の変数に代入するということです。この代入文とC言語での(代入演算子による)代入とでは何が異なるのでしょう。それは＝という操作が値を持つか否かの違いです。C言語の＝は演算子ですから演算結果（式の値）を持ちます。それは＝の右辺の値そのものです。一方，代入文では（式ではない）文の値などというものは意味がありません。C言語で代入という操作が値を持つことには少なくとも2つの意義が考えられます。

まず，第1は複数の変数に1つの文（式）で同じ値を与えることができます。

a＝b＝c＝0；

という式は，表2の結合規則（右にある＝から先に結合される）から，

a＝(b＝(c＝0))；

と同じです。これは0を変数cに代入し，その演算結果である0を変数bに代入し，その演算結果である0を変数aに代入するという記述です。結局は変数a，b，cに0を代入します。これは3つの文で記述する，

a＝0；b＝0；c＝0；

と同じことですが，簡潔さという点では，

a＝b＝c＝0；

のほうが優っていますね[3]。

代入操作が値を持つことの第2の意義は，代入した値をすぐ参照できるということです。これは特に，while文やif文などの条件指定で使用すると効果的です。

while(a[i]＝b [i]) i＝i＋1；

は配列bに格納された文字列を配列aにコピーするためのプログラムです。ここではb [i] の値として0をa [i] に代入したら繰り返しが終了します。ここでは，代入と同時にb[i]の値をテストしています。もし，代入とb[i]のテストを別にするなら，

do {a[i]＝b[i]； i＝i＋1；} while(b[i－1])；

あるいは，

```
while(b[i]) {
  a[i]＝b[i]； i＝i＋1；
}
a[i]＝b[i]；
```

となるでしょうか。どちらにしても何か中途半端なプログラムになってしまいますね。また，簡潔さという点でも，代入と同時にb[i] をテストするほうが勝っています。

どちらの意義においても簡潔さという点が目立ちますが，実際C言語では式を簡潔に記述するためにありとあらゆる記法が用意されているように思います。代入演算子には＝のほかにも，

**＋＝**
**－＝**
**＊＝**
**／＝**

などの，

**演算子＝**

という形式の演算子があります。

式1　演算子＝　式2

という式は単純に，

式1 ＝ 式1　演算子　式2

という式の省略形です。つまり，

x＋＝y－1

は，

x ＝ x＋(y－1)

と同じです。はっきりいってここまで省略する必要があるのっていう気もしますが，

a[i＋j] [b＊c＋4] [i＋2] ＝
　a[i＋j] [b＊c＋4] [i＋2]＋b[i] [j＋c] [10]

などという複雑な式よりも，

a [i＋j] [b＊c＋4] [i＋2] ＋＝ b [i] [j＋c] [10]

のほうが見やすいのは確かです（書き間違いも少ないでしょう)。なお，K＆Rでは代入演算子の利点として「人間の考えに一致している」という点を挙げています。「iに2を加えろ」とか「iを2ずつ増やせ」という表現のほうが「iを取ってきて2を加え，iに書き戻せ」よりも人間の思考に近いことを理由に，

i＋＝2；

のほうが，

i＝i＋2；

よりも好ましいとしているのです。いわれてみればそうですけどね。

**●インクリメント/デクリメント演算子**

C言語のプログラムで使われる式の中でもっとも頻繁に見つけることのできる演算子がこのインクリメント/デクリメント演算子でしょう。これは，

**＋＋**

あるいは，

**－－**

という演算子です。その機能は変数の値を1だけ増加（＋＋）したり1だけ減少（－－）したりします。プログラムの中には，

n＝n＋1

とか，

n＝n－1

といった記述がよく現れます。このインクリメント/デク

リメント演算子はこれらの表現を簡潔に表現するための演算子です。これらは，インクリメント/デクリメント演算子を使って，

n++　あるいは　++n

とか，

n−−　あるいは　−−n

と表現されます。さらに，これらは代入演算子を使った，

n+=1

あるいは，

n−=1

などとも同じ意味です。

ただし，このような単純な式の置き換えだけの意味しかないなら，インクリメント/デクリメント演算子は必要ありません。この演算子はもっと大きな意味を持っています。以前もいいましたが，C言語は配列やその特殊形である文字列を扱うことが非常に多い言語です。配列は添字で参照しますが，その参照時には添字を1ずつずらして処理を行うことがしばしばです。インクリメント/デクリメント演算子はこのような配列の参照を簡潔に行うために存在する演算子といっても過言ではありません。

たとえば，2つの文字列を連結するstrcatというライブラリ関数をC言語で書き下してみましょう。単純に考えれば片方の文字列の終わり（0という文字）を探して，そこから他方の文字列をコピーするプログラムでよさそうです。これは具体的には，

```
strcat(s1,s2)
char s1[],s2[];
 {
  int i,j;
  i=j=0;
  while(s1[i]) i=i+1;
  while(s1[i]=s2[j]) {
      i=i+1; j=j+1;
  }
 }
```

というプログラムになります。これをインクリメント/デクリメント演算子を使用して書き換えれば，

```
strcat(s1,s2)
char s1[],s2[];
 {
  int i,j;
  i=j=0;
  while(s1[i]) i++;
  while(s1[i++]=s2[j++]) ;
 }
```

となり，結構簡略化されます。これは配列の添字にインクリメント/デクリメント演算子を用いた例ですが，配列の添字以外ではfor文による繰り返しにおいて，

```
for(i=0; i<100; i++) {…}
```

などと，繰り返しの制御変数の値を更新する場合にもこの演算子は使用されるようです。このように，インクリメント/デクリメント演算子はC言語のプログラムで多用されますから，この演算子を使用するだけでなぜかプログラムがC言語らしくなるのです。繰り返しの制御変数の更新などは素直に，

i=i+1

と書いても見やすさには大差ないような気もしますが，C言語のプログラマはついインクリメント/デクリメント演算子を使用してしまうものなのです。

配列の添字の記述に便利というだけではインクリメント/デクリメント演算子の存在意義としてはまだ不十分という気がします。本当の意義は++や−−といった演算子が変数名の左につくか右につくかで式に全然違う意味を持たせることができるということです。左についた場合，変数の値の増減は式の計算の前に行われます。逆に，右についた場合，変数の値の増減は式の計算のあとに行われます。したがって，

(++n)+3

という式と，

(n++)+3

という式では，式の値が異なることになります。すなわち，

(++n)+3

は，

n=n+1; n+3

という意味（〰部が式の値）ですから，n+4が式の値となります。また，

(n++)+3

は，

n+3 ; n=n+1

という意味（〰部が式の値）ですから，n+3が式の値となります。ただし，どちらの式においても式の値を求めた後はnの値が1だけ増加していることには変わりありません。配列の添字にインクリメント/デクリメント演算子をつける場合を考えてみましょう。

a [++n]=10;

という式は，

n=n+1; a [n]=10;

という2つの式を意味し，

a [n++]=10;

という式は，

a [n]=10; n=n+1;

という2つの式を意味します。++が添字の左にあるか右にあるかは，添字を増加してから値を代入するか，値を代入してから添字を増加するかの違いがあります。C言語ではこのように他のプログラミング言語では2つに分けて記述されるべき2種類の式を，インクリメント/デクリメント演算子を変数名の左右につけ分けることによ

って1つの式で表現できるようになっているのです。

なお，++を変数名の左につける場合はプリ・インクリメント，変数名に右につける場合はポスト・インクリメントと呼んでいます。同様に，−−を変数名の左につける場合はプリ・デクリメント，変数名に右につける場合はポスト・デクリメントと呼んでいます。

ところで，配列の添字や式の一部で使用せず，インクリメント/デクリメント演算子をただ1つの変数名に適用する場合（それ以外に演算子のない式）では，その意味に違いはありません。すなわち，

```
++n;
```

と，

```
n++；
```

はまったく同じものです。for文による繰り返しにおいては，先の例のようにポスト・インクリメントの形式が多いように思いますが，

```
for(i=0； i<100； ++i) {…}
```

などと，プリ・インクリメントの形式で記述する人もいるようです。このような場合，プリ・インクリメントとポスト・インクリメントのどちらにするかは個人の好みでしょう[4]。

なお，このインクリメント/デクリメント演算子は副作用として値を変更する演算子ですから，この演算子を適用できる対象は，代入演算子の左辺になれるものと同じです。すなわち，++や−−は変数（名）や配列要素などにしかつけることができません。

**●ビットを操作する演算子**

C言語で特徴的な演算のひとつはビットの操作です。これは整数値を2進数のビットパターンとみなして，その各ビットごとに論理演算を行います。2進数はパソコンの入門書などの冒頭でよく紹介されている0と1からなる数ですが，すべての整数が2進数で表されることはわかりますね。C言語では整数を2進数で表したときの0と1のパターンに対し，演算を行うことができるようになっているのです[5]。

演算の種類には論理積（&），論理和（|），排他的論理和（^），否定（~），左シフト（<<），右シフト（>>）の6種類が用意されています。ただしこれらのビット操作は，OSやアセンブラやコンパイラなどの基本プログラムをC言語で記述する場合（C言語をアセンブリ言語の代用にする場合）にはよく使われますが，普通のプログラム（C言語を高級言語として使う場合）ではあまり見掛けません[6]。せいぜい，整数値の上位何ビットかを0にするための，

```
n= x & 0x000000ff;
```

などという記述を見るくらいでしょう。とはいえ，K&Rの中ではビット操作を利用する練習問題が多く載っているので，実際は大事な演算なのでしょう。しかし，この連載は一応高級言語としてのC言語入門ということになっていますから，ここではビット操作にこれ以上立ち入るのはやめておきましょう（そのうち，本格的に説明することがあるかもしれないけど）。

**●条件式**

条件式も式を簡潔に書くために用意されている記法です。式を記述するとき，ある条件の成立/不成立に従って式の値を変更したい場合があります。たとえば，変数xと変数yの最大値を変数zに代入する場合を考えましょう。これは，

```
x>y
```

という条件が満たされるときzにxを代入し，そうでないときzにyを代入することになります。単純に考えればif文を使用して，

```
if (x>y) z=x; else z=y;
```

というプログラムになります。これは2つの文（if文とelse文）を使っていますが，条件式を用いれば1つの文で済ますことができます。条件式は，

**式1　?　式2　:　式3**

という形式で使用します。その値は，式1の値が0でないとき式2の値，式1の値が0のとき式3の値になります。式1には通常等号や不等号（等値演算子や関係演算子）を使った式が記述されます。その値が0でないということは，式1の条件が成立するということです（もちろん0なら不成立）。そこで，xとyの最大値をzに代入する場合は，

```
z=(x>y) ? x : y;
```

というプログラムになります。このように，ある条件に対する値の選択程度の仕事なら，わざわざif文やswitch文といったものものしい制御構造のお世話にならなくても，条件式で簡潔に記述することができるのです。条件式の値を返すif文と思っていてもいいでしょう。

ここで，?の左の条件を示す部分にカッコがついていますが，? や：という演算子と他の演算子の優先順位を比べる（表2を見てね）と，?や：の優先順位はかなり低い（かろうじて代入演算子とカンマ演算子よりも高いだけ）なので，カッコはほとんどすべての場合で不要です。ただし，条件の式がカッコをつけることによって，式の全体が見やすくなるのでカッコをつけるようにしましょう（とK&Rにも書いてある）。

ところで，ここで注意しておくことは条件式は式に変わりないということです。?，：という二段構えの演算子を使用していますが，条件式の，

```
(x>y) ? x : y
```

という部分は形式上は（あるいは気分的には），

```
x + y
```

などというごく普通の演算の記述と同じものです。つまり，条件式を他の式の一部や関数への実引数に持ってきて，

```
z=((x>y) ? x : y)*3 + ((x==y) ? 1 : 0)；
```

や,

func((x>y) ? x : y, z);

などという記述をすることもできるのです。

●**カンマ演算子**

カンマ演算子はなかなか興味深い演算子です。カンマ(,)の左側にある式を計算したあと，カンマの右側にある式を計算し，カンマの右側にある式の値をカンマ演算子の値（演算結果）とします[7]。

x=(a+1,a+2);

という式では変数xにはカンマの右にあるa＋2の値が代入されます（全体にカッコがついているのはカンマ演算子のほうが代入演算子よりも優先順位が低いため)。カンマの左の式は計算はされるのですが，その値は最終的に無視されてしまいます。そのため，カンマの左側は代入演算や関数呼び出しなどの副作用が発生するような式でないとまったく意味がありません。

カンマ演算子を複数含むような式では，カンマで区切られたいくつかの式を次々に計算していき最後に計算した式（いちばん右側にある）の値をその全体の式の値とします。たとえば,

a=a+2, b=c+d, d=3, e=f+10;

という式は左から順に変数aに2を加え，変数cと変数dの値を加えて変数bに代入し，変数dに3を代入し，変数fの値に10を加えた値を変数eに代入した後，最後に計算したf＋10という値をその演算の値とします。つまり,

x=(a=a+2, b=c+d, d=3, e=f+10);

などという式では，変数xにf+10の値が代入されることになります。

a=a+2, b=c+d, d=3, e=f+10;

という式の場合，式を4つに分けて,

a=a+2; b=c+d; d=3; e=f+10;

と記述すればいいと思われるかもしれません。それはもっともです。if文なども,

if(a==10) b=12, a=100;

と記述するよりも,

if (a==10) {b=12; a=100;}

としたほうがはるかに素直ですね。しかし，C言語の文法では1つの式を書くことしかできないのに{}を用いる複文を使えないことがあります。たとえばfor文で初期設定，終了条件，制御変数の増分を指定するためのフィールドなどです。このフィールドに2つ以上の式を書く必要があるとき，カンマ演算子は威力を発揮します。たとえば,

for(i=0, j=0; i+j<100; i++, j++) {…}

というfor文は初期値として変数iと変数jに0を代入し，各ループの終わりでiとjの値を1増やすということを意味します。変数iも変数jもループに関わる制御変数であるとしたら，そのどちらかの変数をfor文の初期

**表1　演算子による式の分類**

| 演算子の分類 | 記号 | 意味（演算後の値） |
|---|---|---|
| 加法演算子 | +<br>- | 左辺と右辺の値の和<br>左辺と右辺の値の差 |
| 代入演算子 | =<br>+=<br>-=<br>*=<br>/=<br>%=<br>&=<br><<=<br>>>=<br>\|=<br>^= | 右辺の値そのもの<br>左辺と右辺の値の和<br>左辺と右辺の値の差<br>左辺と右辺の値の積<br>左辺と右辺の値の商<br>左辺と右辺の値の剰余<br>左辺と右辺の値のビットごとの論理積<br>左辺の値を右辺の値だけ左シフトした値<br>左辺の値を右辺の値だけ右シフトした値<br>左辺と右辺の値のビットごとの論理和<br>左辺と右辺の値のビットごとの排他的論理和 |
| ビット演算子 | &<br><<<br>>><br>\|<br>^ | 左辺と右辺の値のビットごとの論理積<br>左辺の値を右辺の値だけ左シフトした値<br>左辺の値を右辺の値だけ右シフトした値<br>左辺と右辺の値のビットごとの論理和<br>左辺と右辺の値のビットごとの排他的論理和 |
| キャスト演算子 | ( ) | 右辺の値を( )内のデータ型に変換した値 |
| 等値演算子 | ==<br>!= | 左辺と右辺の値が等しいと1，それ以外で0<br>左辺と右辺の値が等しいと0，それ以外で1 |
| 論理演算子 | &&<br>\|\| | 左辺と右辺の値の論理積<br>左辺と右辺の値の論理和 |
| 乗法演算子 | *<br>/<br>% | 左辺と右辺の値の積<br>左辺と右辺の値の商<br>左辺と右辺の値の剰余 |
| 後置演算子 | [ ]<br>( )<br>.<br>-><br>++<br>-- | [ ]内の添字で左辺の配列を参照した値<br>( )内の引数で左辺の関数を呼び出した値<br>左辺の構造体/共用体を右辺のメンバで参照した値<br>ポインタで示される左辺の構造体/共用体を右辺のメンバで参照した値<br>左辺の変数の値，参照後に変数値を1だけ増加<br>左辺の変数の値，参照後に変数値を1だけ減少 |
| 基本式 | 変数名<br>定数値<br>文字列<br>( ) | 変数の値<br>定数の値そのもの<br>文字列が格納されたアドレス値<br>( )内の式の値 |
| 関係演算子 | <<br><=<br>><br>>= | 左辺の値が右辺の値より小さいと1，それ以外で0<br>左辺の値が右辺の値より大きいと0，それ以外で1<br>左辺の値が右辺の値より大きいと1，それ以外で0<br>左辺の値が右辺の値より小さいと0，それ以外で1 |
| 単項演算子 | &<br>~<br>*<br>+<br>-<br>!<br>++<br>--<br>sizeof | 右辺の変数のアドレス値<br>右辺の値の1の補数（ビットごとの論理否定）<br>右辺のポインタ変数が示す値<br>右辺の値の符号を保存した値(右辺の値そのもの)<br>右辺の値を符号反転した値<br>右辺の値が0なら1，それ以外で0(論理否定)<br>右辺の変数の値を1だけ増加し，その値<br>右辺の変数の値を1だけ減少し，その値<br>右辺の( )内のデータ型の大きさ(バイト数) |
| 3項演算子 | ? : | ?の左辺の値が0なら：の右辺の値，それ以外なら：の左辺の値 |
| その他 | , | 左辺の値を計算した後，右辺の値 |

（注）代入演算子では，副作用として演算結果を左辺の変数に代入する。

設定フィールドや増分指定フィールドの外に追いやるのはプログラムをわかりにくくするだけです。もし，このfor文が，

　　j＝0；

　　for(i＝0； i＋j＜100； i＋＋) {…； j＋＋}

などと記述されていたら，変数jもループに関係するということに気付く人が何人いるでしょうか。カンマ演算子はあまり意味のない演算子のようにも思えますが，使うべきところに使えばプログラムを見やすくする効果が出るのです。

**●論理演算子**

論理演算子は等値演算子や関係演算子で指定される式を結合するための演算子です。&&はAND条件，||はOR条件です。たとえば，

　　ch＞＝'A' && ch＜＝'Z'

という式は，

　　ch＞＝'A'

という式と，

　　ch＜＝'Z'

という式がどちらも成立する（値が0でないこと）ときに1を値とし，そうでないときに0を値とします。逆に，

　　ch＞＝'A'　||　ch＜＝'Z'

という式では，

　　ch＞＝'A'

という式か，

　　ch＜＝'Z'

という式のどちらかが成立する（値が0でないこと）ときに1を値とし，そうでないときに0を値とします。と，このような説明はしなくてもわかりますね。こんなわかりきったことはおいといて，ここでは制御構造としての論理演算子について説明しましょう。つまり，&&や||という演算子はif文やswitch文としての性質も有しているのです。

　　**式1　&&　式2**

という式を考えましょう。&&は論理のANDを取る演算子ですから，この式の値を決定するためには式1と式2の値を求めなくてはなりません。式1か式2のどちらかの値が0ならばこの式の値は0であり，式1と式2がどちらも0でなければこの式の値は1です。と，普通の人は考えるでしょう。

しかし，C言語では上の式はこのように解釈されません。C言語では，まず式1の値を求めてみて，それが0でないとき初めて式2の値を求めにいきます。式1の値が0なら全体の式の値は0であることがわかるので式2の計算は行われないのです。そして，全体の式の値は式2の値が0でないとき1となり，そうでないときは0となります。これは，まさに式1を条件とした制御構造です。つまり，

　　式1　&&　式2

という式は，意味的には，

　　(式1)？ 式2　：　0

という条件式とほぼ同じです。一方，論理のORである||では&&とは条件の式の値の0と1が逆転していると考えればいいでしょう。つまり，式1の値が0であるときにのみ式2の値を求めにいきます。式0の値が0でないなら全体の式の値は1であることがわかるので式2の計算は行われません。そして，全体の式の値は式2の値が0のとき0となり，そうでないときは1となります。

　　式1　||　式2

を条件式で表せば，

　　(式1)？　1　：　式2

となるでしょうか。C言語の論理演算と一般的なプログラミング言語における論理演算との差は，演算子の右にある式2を計算するか否かです。一般的にはどちらの場合も同じ結果になるので，その差を意識する必要はありません。しかし，式2に副作用がある場合は式2を計算するか否かで結果が異なってきます。たとえば，

　　if((n＝＝1)　||　(＋＋n＞3)) {…}；

というif文で使われる論理演算では変数nの値が1でないときのみ，||の右の式の中の＋＋nによってnの値が1だけ増加します。もし，||の両側の式をつねに計算するのであれば，nがどのような値であっても値が1増加するはずです。

以上のように論理演算子は制御構造を持っていますから，それをif文や条件式のように使用することもできます。たとえば，

　　fact(n)

**表2　演算子の優先順位と結合規則**

| 演算子 | 結合規則 |
|---|---|
| ( )　[ ]　->　. | 左から右 |
| !　~　++　--　+　-　*　&　(type)　sizeof | 右から左 |
| *　/　% | 左から右 |
| +　- | 左から右 |
| <<　>> | 左から右 |
| <　<=　>　>= | 左から右 |
| ==　!= | 左から右 |
| & | 左から右 |
| ^ | 左から右 |
| \| | 左から右 |
| && | 左から右 |
| \|\| | 左から右 |
| ?: | 右から左 |
| =　+=　-=　*=　/=　%=　&=　^=　\|=　<<=　>>= | 右から左 |
| , | 左から右 |

（注）表の上にある演算子ほど優先順位が高い（先に計算される）。
同一行の演算子は同じ優先順位を持つ。

```
int n;
  {
    int r;
    return((r=(n==1)) || (r=n*fact(n-1)), r);
  }
```

は論理演算子を使用した階乗計算のプログラムです。論理演算子の結果は0か1でしかないので，演算子の両辺にある式の値を記憶するために補助的な変数rを導入し，その値をカンマ演算子で返すというむちゃくちゃなことをやっていますが，これでも正しく動きます。

関数の値が0か1でしかない場合はカンマ演算子などという姑息な手を使わなくても論理演算子によって簡潔なプログラムが書けるようになると思います。ただし，論理演算子を制御構造として積極的に利用したり，論理演算子の式の中（特に演算子の右辺）に副作用を生じるような式を書くことは，式自体が非常に理解しにくくなるだけでなく，バグが入る要因にもなりますから，あまり勧められることではありません。

---

2) とはいえ，他人の書いたプログラム中の式を理解するためには演算子の優先順位を知っておく必要がある。その昔，C言語を学ぶ者は演算子の優先順位表を机の前に貼り，毎日それを眺めては九九のように暗唱していたという（ウソ度90%）。

3) 最近の拡張されたFORTRANの処理系では，

A=B=C=0

などという記述を許すものもある。

4) 私はポスト・インクリメントの形式を多用しているので，プリ・デクリメントを使用したプログラムを見ると新鮮な気分になる。

5) 確かにビットの操作を提供するプログラミング言語はC言語以外には少ない。このようなビット操作は本来はアセンブリ言語の守備範囲である。ただし，BASICには論理積，論理和，排他的論理和，否定程度のビット操作は必ず備わっている。

6) C言語ではOSやアセンブラを記述するほうがむしろ普通のような気がする（だからこそビット操作の意味がある）が。

7) C言語の特徴的な演算子を見てLISPとの類似性に気づいた人もいると思う。代入操作（SETQ）や条件選択（COND）は値を返してくるし，カンマ演算子はPROGN関数そのものである。それ以前にC言語もLISPも関数を主体とするプログラミング言語である。

### ◆基礎力を高めよう

**設問1**　a+++bという式は次の2通りに解釈可能ですが，実際のCコンパイラではどう解釈されるか調べてください。

1) (a++)+b
2) a+(++b)

**設問2**　次に示す式はCコンパイラでは文法違反になりますが，1組の（ ）で式の一部を基本式とすれば文法違反でなくなります。1組の（ ）をつけて正しい式にしてください。

1) a++b
2) a++++b
3) a+++++b

**設問3**　次に示す式に誤りがあれば指摘してください。ただし，変数xと変数yはint型，変数dはdouble型とします。

1) +x=1；
2) −x=1；
3) x++=y;
4) +++d;
5) ++(+d);
6) (x=y)++；
7) x=d%y;
8) x=(d<=y);
9) y=d&x;
10) y=d<<x;

（解答は下）

## おわりに

C言語における式は簡潔さをモットーとしています。式を簡潔に表現するためにC言語では他のプログラミング言語では見られないような演算子がたくさん用意されています。C言語で書いたプログラムは一種独特な雰囲気を持っていますが，それは演算子の外見によるところが多いのです（++なんていう演算子を使うだけで気分はもうCプログラマ！）。英語などの外国語を上達しようと思うなら，その外国語で考えるようにならなければだめだとよくいわれます。C言語もそれと同様で，普段からC言語の文化に慣れ親しんでこそ上達が見込めるというものです。それにはまず，C言語の簡潔な演算子を使いこなしてゆくことから始めるのがいいでしょう。

さて，次回は標準入出力とリダイレクトに関して解説したいと思います。それでは，来月までさようなら。

### ◆基礎力を高めようの解答

**設問1**

1)の式のように解釈される。

**解説**

おそらくすべてのCコンパイラがそう解釈する。なぜ，といわれても困る。

**設問2**

1) a+(+b)　2) a+++(+b)　3) a+++(++b)

**解説**

符号を保持する単項演算子+はANSI Cで導入された。したがって，ANSI Cに非準拠のXCのバージョン1では1)，2)のような表現はできない。3)に関しては(a++)+++bであっても一見は正しそうである。しかし，この式が((a++)++)+bと解釈されること（設問1を参照）を考えると，a++をポスト・インクリメントしているので，++を適用できるのが変数名または配列要素という規則に違反する。

**設問3**

1)，4)，5)，8) 正しい。

2)，3) =の左辺が不正。

6)++を適用する対象が不正。

7)浮動小数点の剰余は求められない。%は整数のみに対して適用できる。

9)，10)ビット操作の演算は整数にのみに対して適用できる。

**解説**

1)，5)：(単項演算子の)+は不思議な演算子である。Cコンパイラでは，+が付いた式でも+が付いてないものと解釈しているようである。本来なら1)や5)は許されないはずであるが，GCCでもXC(バージョン2）でもエラーにはならない。

7)：IEEEというアメリカの学会の規格では，

x % y = x−y*int(x/y)

という関係によって浮動小数点の剰余が定義されている。

## X1用ゲーム

# The Master of Payment

Shibata Atsushi
柴田　淳

Rythm to Trace, MUD BALLIN'などの独創的なプログラムでお馴染みの柴田氏がOh!Xスタッフとして登場。今回は買い物ゲーム(?)で美しい財布のあり方を追求します。あなたも支払いの達人となって経済的カタルシスを味わってみませんか。

僕の親戚に，いつもポケットに一杯小銭を入れて，じゃらじゃらと音を立てながら歩く人がいます。そしてその人は子供と見ると決まって「重たいからあげるよ」といって，その両手に一杯の小銭をくれるのです。僕が小さい頃は，正月や盆に田舎に行くと，そのじゃらじゃらがいつ聞こえてくるかと期待に胸を膨らませたものでした。

両手に一杯といってもたいていは五円玉や一円玉で，全部合わせてせいぜい1,000円くらいにしかなりませんでしたが，当時の僕にとってはずいぶん大きな臨時収入でした。そしてあるとき，どうしていつもこんなに小銭をためているのだろうかと不思議に思って，その理由を聞いてみたことがありました。

「だってお金がいっぱいあると，お金持ちの気分が味わえるじゃないか」

僕はその答えになんだか納得できず，一円玉が1万個よりも，一万円札が1枚のほうが高額に見えるのではないかというような意味のことを聞き返しました。するとその叔父さんはくすくす笑って，それならその小銭と千円札を交換するかと僕に尋ねたのです。そのときはまだ，もらった小銭が合計いくらになるのか数えていなかったので，はたして交換するほうが得か，それともそのまま持っていたほうが得かと悩んだ挙句，僕は「取り替えなくていい」とぼそっといいました。なんとなく小銭のほうが多いような気がしたのです。

そのあとすぐに物置に駆け込んでもらった小銭を勘定すると，なんと1,000円より少ない額しかありませんでした。僕はそのとき，子供心になんともいえない複雑な悔しさを味わったのを覚えています。自分で小銭より札のほうが高く見えるなどといっておきながら，実際比べてみると小銭のほうが多い気がして，その心移りの結果損をする羽目になったのです。叔父さんもからかうつもりで小銭と千円札を交換しようなんていいだしたのでしょうし，そう考えると余計に悔しくなりました。

そんなことがあって以来，僕は財布には最低限の小銭しか入れておかないようになりました。小銭がたくさんある財布を見ると，どうしてもあのときの悔しさが思い出されてならないのです。

## 支払いの達人

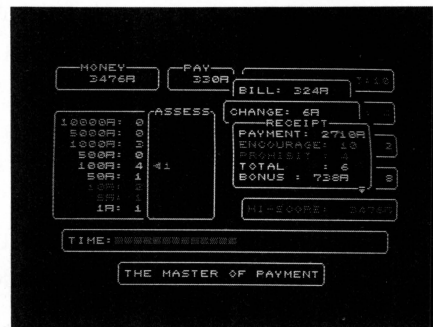
さて，ここで便宜上最低限の小銭しか持たない財布を仮定して，これを純粋理性的財布と呼びましょうか。逆に不要な小銭でいっぱいの財布を，不純野性的財布としましょう。最低限の小銭とはなにか，ということは，そろばんの玉を思い浮かべればわかると思います。一の玉は4個で，五の玉は1個しかなく，しかしその個数を超えることなしに，すべての数を表すことができるのです。とすれば十円玉は4個以上は不必要ですし，五百円玉だって1個でこと足りる，ということになります。最小限の小銭とは，それ以上あれば桁上がりを余儀なくされる個数以下の小銭，とでもいい換えられるでしょうか。

僕はたまたま個人的な経験があり，最低限の小銭だとか純粋理性的財布などといった馬鹿げたことを考えているのですが，世の中のたいていの人は買い物のたびにそんなことを考えているほど閑ではないでしょう。小銭が増えないように気を使っていると自認する人でも，せいぜい金額の末尾をぴったり払うくらいではないでしょうか。

その点僕の支払い方は，ときには少々奇妙ですらあります。そのせいか，僕が買い物をして支払いをすると，レジの人にびっくりしたような顔で見つめられることがあります。

あるとき，僕はスーパーで867円の買い物をしました。財布の中身を見ると，ぴったり払うには小銭が足りません。こういうときこそ，僕の財布が純粋理性的財布であり続けるか，それとも不純野性的財布に堕落してしまうかの瀬戸際なのです。僕は財布の残金をもう一度確かめ，少しのあいだ天井に視線を走らせ，おもむろに1,422円取り出し，レジのおばさんに渡しました。

一見するとこの2つの数字にはなんの関連性もなく，この金額を払ったなら，むしろたくさんおつりをもらうことになるのではないかという印象さえ受けるでしょう。しかしそこが僕の財布が純粋理性的であり，あなたの財布が不純野性的である所以なのです。そのときのレジのおばさんもやはり，けげんそうな顔で僕の差し出したお金を受け取り，「なんだろねぇ，この人は」とでもいいたそうに僕を見て，レジに金額を打ち込んでいました。

ところが，555円というおつりの表示を見たとたん，いままでの変人でも見るかのような目つきは，一転して羨望の眼差しに変

わりました。好きで数学を学んでいる人に，数学のどこが面白いのかと質問すると，美しい式に出会うのが楽しいのだという答えが返ってくるときがあります。とても複雑な式が，慎重に展開していくと最終的に対称形を成したり，意味あり気な係数が並んだりという場面には，受験勉強で数学をやった人なら何回かは遭遇したことがあるのではないでしょうか。そんなときの脳味噌の奥のほうから染み出してくるような喜びが，代金を払う僕には当然のこと，このときのレジのおばさんにも感じられたようです。

たまにこういった美しいつり銭が現れるので，あるときなどは魚屋のおばさんに「おみそれしちゃったわよ」といわれたことがあったし，このときはこのときで，レジのおばさんは僕の顔をまじまじと見つめて「すごおいっ！」といったあと，ほかのお客さんのいる目の前で拍手までしてくれました。

## この感動をあなたに

日本国勢図会という統計資料があるのを知っているでしょうか。ついこのあいだその1990年版をなにげなく見ていると，日本の貨幣の流通量という項目が目に止まりました。それによると日本中には十円玉が189億個，五円玉は100億個，一円玉はなんと300億個も流通しているのだそうです。

この人口の何百倍もの小額貨幣は，あるいは貯金箱に入れられて欽ちゃんに手渡され，あるいは道端で寂しく風雪にさらされているものもあるでしょう。しかし僕がこの数字を見て思い浮かべたのは，財布にはち切れんばかりに詰め込まれ，じゃらじゃらとひしめきあっている小銭の姿でした。

あなたも今日から，こんなふうに醜い財布とはおさらばして，簡素で知的な生活を志してみてはいかがでしょうか。

**評価基準**

| 評価対象となる事項 | 評価点 |
| --- | --- |
| ・硬貨のうちのどれかをなくす | ＋１ |
| ・必要以上の硬貨，または紙幣をためる（たとえば十円玉なら５個以上，五千円札なら２枚以上など） | －２ |
| ・所要時間を超える | －２ |
| ・支払った硬貨，紙幣と同じものがおつりとして戻ってくる | －１ |
| ・代金に満たない金額を誤って支払う | －１ |

## ゲームについて

このゲームはなるべく小銭を増やさないように買い物をしていくというゲームです。１回買い物を済ますごとにコンピュータがその支払いの態度の「善し悪し」を評価し，それによって今後の展開が決まるのです。

プログラムはすべてBASICで書かれています。ただしウィンドウ処理に一部マシン語を使用していますので，入力が終わったら，2170行から2280行を念入りにチェックし，走らせる前に一応セーブしておいたほうがいいでしょう。

## ゲームの進め方

このゲームは，８回の買い物でひとつのレベルが成り立っています。またレベルに応じて制限時間が割り当てられ，当然レベルが上がるにつれその時間も短くなります。

まず画面の説明をしましょう。左側に縦に２つ並んでいるウィンドウは，現在の所持金額とそれぞれの貨幣が何個あるかを示すものです。その右のものは，支払う金額と各貨幣の枚数を示しています。

初めの持ち金は千円札が２枚，つまり2,000円です。ゲームを開始すると，画面の右上にその回の代金が表示されます。この金額と自分の持っている貨幣とを見比べて，臨機応変に支払いをしてください。

支払いの方法は，貨幣の個数を表示するウィンドウの中の三角形のカーソルを，８と２のキーで上下に動かし，払いたい貨幣の段まで持っていきます。

そこでＺキーを押すと，自分の持ち金から支払い分に目的の貨幣が加算されます。またＸキーを押すと，逆に支払い分から貨幣を引き上げます。この操作を払いたい貨幣の種類だけ繰り返し，目的の金額に達したらリターンキーを押してください。それで１回の買い物は終了です。

また，ゲームの途中で進行が止まっているときがあると思いますが，そのときはウィンドウの右下に下向きの三角形があるはずです。これはRPGにありがちなキー入力をうながすマークですので，これが出てきたらなにかキーを押してください。

買い物が終了するとおつりが加えられ，そのあと支払いの態度を評価してくれます。評価基準は表にまとめておきましたので，そちらを見てください。画面にはプラスの評価は緑で表示され，マイナスの評価は赤で表示されます。

さて，この評価点はその都度の買い物で進行に影響を及ぼすわけではありません。８回の買い物中，プラス点とマイナス点はどんどん加算されていきます。

そして８回終わった時点で，プラス点がマイナス点を上回っているか，もしくは同じであればボーナスが加算され，上のレベルに挑戦することができます。

しかしマイナス点のほうが多い場合には，そこでゲームは終了です。

## リスト

```
1000 '   -- THE MASTER OF PAYMENT --
1010 '                    A.SHIBATA   1990
1020 CLEAR &HF000:GOSUB 2040:POKE &HF0DA,0,&HF1:HI%=2000
1030 GOSUB 2020:GOSUB 1790:GOSUB 1710
1040 ' MAIN
1050 FOR I%=0 TO 8:P%(I%)=0:NEXT:LV%=1:P%=2000:P%(2)=2:F%=0
1060 GOSUB 1710:A$="LEVEL"+STR$(LV%):GOSUB 1900
1070 CA%=1:EC%=0:PR%=0:GOSUB 1240:X%=20:Y%=8:A%=14:B%=5:GOSUB 1940
1080 LOCATE 24,8:PRINT"RECEIPT":LOCATE 21, 9:PRINT USING"PAYMENT:#####円",Q%(0)
1090 A%=Q%(0):GOSUB 1660:COLOR4:LOCATE 21,10:PRINT "ENCOURAGE:";EC%:COLOR 2
1100 LOCATE 21,11:PRINT "PROHIBIT :";PR%:COLOR 7:LOCATE 21,12:PLAY "R9"
1110 PRINT "TOTAL     :";:IF EC%<PR% THEN 1180
1120 PRINT EC%-PR%:LOCATE 21,13:PRINT "BONUS :";STR$((EC%-PR%)*123);"円"
1130 A%=(EC%-PR%)*123:GOSUB 1660:A%=P%:P%=0
1140 FOR I%=0 TO 8:P%(I%)=0:NEXT:GOSUB 1710
1150 GOSUB 1660:LOCATE 34,14:PRINT "_":IF HI%<P% THEN HI%=P%:GOSUB 1710
1160 IF INKEY$="" THEN 1160 ELSE PLAY M$(0):LV%=LV%+1:A%=10:GOSUB 1990:GOTO1060
1170 ' GAME OVER
1180 COLOR 2:PRINT EC%-PR%:PLAY M$(3):X%=12:Y%=13:A%=15:B%=3:GOSUB 1940
1190 LOCATE 16,14:PRINT "GAME OVER":LOCATE 15,15:PRINT "LEVEL:";LV%
1200 LOCATE 15,16:PRINT "SCORE:";STR$(P%)+"円":LOCATE 27,17:PRINT "_":KEY 0,""
1210 I$=INKEY$:IF I$="" THEN 1210 ELSE IF I$="E" THEN CLS:END
1220 A%=10:GOSUB 1990:GOTO 1030
1230 ' QUESTION
1240 A%=P%:FOR I%=1 TO 4:B%=P%¥9+RND*P%¥9:Q%(I%)=B%:A%=A%-B%:NEXT
1250 C%=A%:FOR I%=5 TO 7:B%=C%¥7+RND*C%¥7:Q%(I%)=B%:A%=A%-B%:NEXT
1260 Q%(8)=A%¥2+RND*A%¥2:Q%(0)=P%-A%+Q%(8)
1270 FOR I%=1 TO 8:SWAP Q%(I%),Q%(INT(RND*8+1)):NEXT
```

```
1280 ' CASHER * 8
1290 GOSUB 1710:A$="CASHER"+STR$(CA%):GOSUB 1900:BI%=Q%(CA%):KEY 0,"5"
1300 X%=20:Y%=4:A%=12:B%=1:GOSUB 1940:LOCATE 21,5:PRINT "BILL:"+STR$(BI%)+"円"
1310 FOR I%=0 TO 8:C%(I%)=0:B%(I%)=P%(I%):COLOR FNA(I%):LOCATE 14,8+I%:PRINT 0
1320 NEXT:D%=0:TIME=0:X%=2:Y%=18:A%=34:B%=1:GOSUB 1950
1330 TI%=27-LV%*2:IF TI%<4 THEN TI%=4
1340 COLOR 7:LOCATE 3,19:PRINT "TIME:";:COLOR 2:PRINT STRING$(TI%,"^")
1350 B%=INSTR("28ZX5"+CHR$(13),INKEY$):IF B%=0 THEN 1450
1360 LOCATE 11,8+F%:PRINT " ":C%=0
1370 IF B%=1 AND F%<>8 THEN F%=F%+1:GOTO 1440
1380 IF B%=2 AND F%<>0 THEN F%=F%-1:GOTO 1440
1390 IF B%=3 AND C%(F%)<>0 THEN C%(F%)=C%(F%)-1:P%(F%)=P%(F%)+1:C%=1
1400 IF B%=4 AND P%(F%)<>0 THEN C%(F%)=C%(F%)+1:P%(F%)=P%(F%)-1:C%=-1
1410 P%=P%+M%(F%)*C%:D%=D%-M%(F%)*C%:COLOR FNA(F%)
1420 LOCATE 9,8+F%:PRINT USING "##<"+CHR$(28,28)+"##",P%(F%),C%(F%)
1430 COLOR 7:LOCATE 4,4:PRINT USING "#####円"+CHR$(28,28,28,28)+"#####円",P%,D%
1440 COLOR 7:LOCATE 11,8+F%:PRINT "<":PLAY M$(B% ¥ 3):IF B%=6 THEN 1490
1450 IF TIME=E% THEN 1350
1460 TI%=TI%-1:E%=TIME:LOCATE 9+TI%,19:PRINT" ";:IF TI%<>-1 THEN 1350
1470 PLAY M$(3):PR%=PR%+2:GOSUB 1710:TI%=26-LV%*2:IF TI%<4 THEN TI%=4
1480 COLOR 2:LOCATE 8,19:PRINT STRING$(TI%,"^"):GOTO 1350
1490 IF BI%>D% THENPLAYM$(3):A$="SHORT!":GOSUB1900:PR%=PR%+1:GOSUB1710:GOTO1350
1500 X%=19:Y%=6:A%=14:B%=1:GOSUB 1940
1510 LOCATE 20,7:PRINT "CHANGE:"+STR$(D%-BI%)+"円":A%=D%-BI%:GOSUB 1660
1520 X%=11:Y%=7:A%=6:B%=9:GOSUB 1940:LOCATE 12,7:PRINT "ASSESS"
1530 ' ASSESSMENT
1540 FOR I%=1 TO 8:A%=0
1550 IF (M%(I%)*B%(I%) ¥ M%(I%-1))<(M%(I%)*P%(I%) ¥ M%(I%-1)) THEN A%=-2
1560 IF C%(I%)<>0 AND A%(I%)<>0 THEN A%=A%-1
1570 IF P%(I%)=0  AND B%(I%)<>0 AND I%>2 THEN A%=1
1580 IF A%=0 THEN 1620
1590 COLOR 3+SGN(A%):LOCATE 12,8+I%:PRINT USING"<#";ABS(A%)
1600 IF A%>0 THEN EC%=EC%+A% ELSE PR%=PR%-A%
1610 PLAY M$(3+(A%>0)):GOSUB 1760
1620 NEXT:GOSUB 1710:A$="PUSH":GOSUB 1900
1630 IF CA%=8 THEN RETURN ELSE CA%=CA%+1
1640 A%=10:GOSUB 1990:GOTO 1280
1650 ' ADDING
1660 FOR I%=0 TO 8:B%=A% ¥ M%(I%):A%(I%)=B%:A%=A%-M%(I%)*B%:NEXT
1670 FOR I%=0 TO 8:FOR J%=1 TO A%(I%):P%(I%)=P%(I%)+1:COLOR FNA(I%)
1680 LOCATE 9,8+I%:PRINT USING "##",P%(I%);:PLAY M$(0):P%=P%+M%(I%)
1690 COLOR 7:LOCATE 4,4:PRINT USING"#####円",P%:NEXT:NEXT:RETURN
1700 ' STATUS
1710 FOR I%=0 TO 8:COLOR FNA(I%):LOCATE 9,8+I%:PRINT USING "##",P%(I%);:NEXT
1720 LOCATE 4,4:PRINT USING "#####円",P%
1730 COLOR 6:LOCATE 36,10:PRINT USING "##",LV%
1740 COLOR 7:LOCATE 36,13:PRINT USING "##",CA%
1750 COLOR 2:LOCATE 32,16:PRINT USING "#####円",HI%
1760 COLOR 4:LOCATE 36, 4:PRINT USING "##",EC%
1770 COLOR 2:LOCATE 36, 7:PRINT USING "##",PR%:RETURN
1780 ' LAY OUT
1790 CLS:RESTORE 1860:FOR J%=1 TO 5:READ X%,Y%,A%,B%:GOSUB 1950:NEXT
1800 X%=21:A%=16:B%=1:FOR J%=1 TO 5:Y%=J%*3        :GOSUB 1950:NEXT
1810 FOR I%=1 TO 5:READ A$,A%:COLOR A%:LOCATE 22,1+I%*3:PRINT A$:NEXT
1820 FOR I%=0 TO 8:COLOR FNA(I%):LOCATE 2,8+I%
1830 PRINT RIGHT$("    "+STR$(M%(I%)),5)+"円:";:NEXT
1840 LOCATE 4,3:PRINT"MONEY":LOCATE 15,3:PRINT"PAY"
1850 LOCATE 9,22:PRINT "THE MASTER OF PAYMENT":RETURN
1860 DATA  1, 3,10, 1, 1, 7,10, 9,13, 3, 6, 1, 13, 7, 3, 9, 8,21,21, 1
1870 DATA"ENCOURAGEMENT:",4,"PROHIBITION  :",2,"LEVEL         :",6
1880 DATA"CASHER        :",7,"HI-SCORE:",2
1890 ' MESSAGE
1900 X%=18-LEN(A$)¥2:Y%=11:A%=LEN(A$):B%=1:GOSUB 1940
1910 LOCATE X%+1,Y%+1:PRINT A$:LOCATE X%+A%,13:PRINT "_"
1920 IF INKEY$="" THEN 1920 ELSE PLAY M$(0):A%=1:GOSUB 1990:RETURN
1930 ' WINDOW OPEN
1940 POKE &HF0D6,X%,Y%,A%+2,B%+2:CALL &HF000:W%=W%+1
1950 CGEN:COLOR 7  :LOCATE X%,Y%   :PRINT "╭"+STRING$(A%,"─")+"╮";
1960 FOR I%=1 TO B%:LOCATE X%,Y%+I%:PRINT "│"+STRING$(A%," ")+"│";:NEXT
1970 LOCATE X%,Y%+I%:PRINT "╰"+STRING$(A%,"─")+"╯";:CGEN 1:RETURN
1980 ' WINDOW CLOSE
1990 FOR I%=1 TO A%:IF W%<>0 THEN CALL &HF054:W%=W%-1:PAUSE1:NEXT
2000 RETURN
2010 ' OPENING
2020 COLOR 7:CLS:A$="THE MASTER OF PAYMENT":GOSUB 1900:RETURN
2030 ' INITIALIZE
2040 WIDTH 40:INIT:LIST 0:PLAY 750:CLICK OFF:CGEN 1
2050 DIM M%(8),P%(8),A%(8),B%(8),C%(8),Q%(8),M$(3)
2060 DEF FNA(A%)=7+(A%<3)+((A%¥2)=3)*5:M$(3)="O3V15E3:O2V15#F3F"
2070 M$(0)="O4V15A0:O4V10F0":M$(1)="O5V15A0:O5V10C0":M$(2)=M$(1)
2080 RESTORE 2170:FOR I%=0 TO 6:READ A$:MEM$(&HF000+I%*32,32)=HEXCHR$(A$):NEXT
2090 FOR I%=0 TO 8:READ M%(I%):NEXT
2100 READ B$:IF RIGHT$(CGPAT$(60),8)=HEXCHR$(B$) THEN RETURN
2110 FOR I%=32 TO 95:A$=LEFT$(CGPAT$(I%),8)
2120 A$=CHR$(0)+MID$(A$,1,2)+MID$(A$,4,1)+MID$(A$,6,2)+CHR$(0,0)
2130 DEFCHR$(I%)=A$+A$+A$:NEXT
2140 FOR I%=1 TO 13:DEFCHR$(ASC(MID$("<_023457SGN^円",I%,1)))=HEXCHR$(B$+B$+B$)
2150 READ B$:NEXT:RETURN
2160 '     1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8
2170 DATA"2AD6F0CDB8F0E511 002819444D2ADAF0 CD2DF0EBE1010030 09444DEBCD2DF0EB"
2180 DATA"21D6F0010400EDB0 EB22DAF0C93AD8F0 573AD9F05FED7877 032315C235F03AD8"
2190 DATA"F057E560693E2892 06004F09444DE11D C235F0C92ADAF02B 11D9F0010400EDB8"
2200 DATA"EB2AD6F03AD9F084 3D673AD8F0853D6F CDB8F0E501003009 444DEBCD8FF0EBE1"
2210 DATA"01002809444DEBCD 8FF02322DAF0C93A D8F0573AD9F05F7E ED790B2B15C297F0"
2220 DATA"3AD8F057E560693E 289206004FB7ED42 444DE11DC297F0C9 C5D506004D6C2600"
2230 DATA"545D292919B7CB15 CB14CB15CB14CB15 CB1409D1C1C9     "
2240 DATA 10000,5000,1000,500,100,50,10,5,1
2250 DATA 000C3CFC3C0C0000,00007C007C381000,003C465A623C0000,003C420C307E0000
2260 DATA 007C0218027C0000,000C14247E040000,007C407C027C0000,007E420408080000
2270 DATA 003E403C027C0000,001C204E221C0000,0062524A46420000,00FEFEFEFEFE0000
2280 DATA 00FE82FE82860000,0
```

## 達人への道

このゲームのこつはなんといっても，ミスを少なくすることにあります。レベルの低いうちは時間もたっぷりありますので，じっくりと考えてどう支払うか決めてください。レベルが上がるにつれて，時間が少なくなっていくだけでなく，ボーナスが加算されることによって払うお金も高くなっていきますので，複雑な操作を要求されるようになります。

僕の場合はだいたいレベル11くらいからミスが目立ってきて，レベル13でもう思考速度がついていかなくなっておしまい，といった感じです。普通の計算能力があれば初めてでもレベル10まではいくことができるでしょう。あとは支払い方のパターンを覚えてしまえば，レベル15くらいまでは達することができるのではないでしょうか。

そしてこのゲームをやって払い方に自信がついたなら，さっそく買い物をするときに実践してみましょう。そういつもいつも先のような美しいおつりに出会うとは限りませんが，一度この快感を味わってしまうと病みつきになること請け合いです。そうなったら最後，もう二度と小銭を払うのが面倒だなどとは思わなくなり，レジに並んでいるときに，1,000円もしない買い物に一万円札を払うおばさんなどを見かけたりすると，無性に腹が立ってくるはずです。

＊　＊　＊

ところでついこのあいだ，例の叔父さんが僕の家に遊びにきました。近くに仕事できたついでだったらしいのですが，相変わらずポケットにたくさんの小銭を詰め込んでいました。

居間でお茶を飲んでいるとき，僕は会話の切れ目を見計らって，今度自分の作ったゲームが雑誌に載るのだと話しました。叔父さんはのっそりと顔を僕のほうに向けて，それはどんなゲームだと聞いてきました。僕は「しめた」と思い，小銭を増やさないように買い物するゲームだ，というと，叔父さんはただ苦笑いするだけでなにもいわず，お茶をずずっとすすりました。

そしてその夜，じゃらじゃらという小銭の音を響かせながら，遠い道のりを帰っていったのでした。

# THE SENTINEL

〈対応機種一覧〉 ●MZ-80K/C/700/1500●MZ-80B/2000●MZ-2500/2861●X1●X1turbo/Z●PC-8001/8801/88●SMC-777/C●PASOPIA/5●PASOPIA 7●FM-7/77/AV●PC-286/386/9801/98●X68000
掲載されたプログラムの利用には各機種用のS-OS"SWORD"システムが必要です。

## 第108部 REAL ソースリスト編

**●REALソースリスト掲載**

ついに巨大なソースリストを一挙掲載することができました。アセンブラで開発されたプログラムとしては最大級の実数型コンパイラREALです。実行ファイルでまるまる16Kバイト，ソースにすると今回のような大きさになってしまいます。

それにしても，毎度のこととはいえ，大貫氏のソースリストの膨大な日本語表記には圧倒されてしまいます。X1turboの漢字変換を使っているのでしょうが，入力の手間だけでどれくらいかかるのでしょうか。

**●Small-C当選者発表**

先月号のこのコーナーで行ったSmall-C ver.2.7ディスク配布希望者募集の当選者を発表します。当選者は以下の50名の方々です（順不同，敬称略）。

桑山直樹（愛知県），高田英基（千葉県），榊卓也（京都府），澤井憲司（大阪府），曽我恭成（福井県），大塚真典（静岡県），澤田保隆（東京都），大島誠（神奈川県），林幸敏（神奈川県），荒洋一（茨城県），桜井清雄（大阪府），石田伯仁（神奈川県），桑原嘉男（東京都），駒井健一（千葉県），常世田一郎（埼玉県），渡辺裕之（北海道），磯部佳成（山口県），高橋淳一（宮城県），山本稔（大阪府），加藤正栄（神奈川県），石田泰弘（兵庫県），西井貴（三重県），難波訓広（神奈川県），山内正人（埼玉県），浦賀毅（茨城県），野村圭一（千葉県），大谷悦正（東京都），栖原紀夫（東京都），小原貴志（埼玉県），谷口利一（千葉県），田窪伸児（大阪府），倉田敏広（神奈川県），上原俊夫（群馬県），伊藤輝之（愛知県），樫正一郎（埼玉県），佐野正文（三重県），中田一隆（秋田県），熊谷正之（千葉県），白井保弘（群馬県），吉田敏幸（神奈川県），飯田茂（東京都），森喜一郎（大阪府），伊藤直也（静岡県），阿部俊光（岩手県），石山伸一（栃木県），勇崎昌宏（京都府），三浦祥吾（北海道），深川哲光（香川県），中林俊一（北海道），伊藤佳夫（埼玉県）

なお，今回のディスク配布当選者の所属する関連サークルは，S-OSユーザーズクラブ，EXTRA，VIP ROOM，Q-System Lab，Illegal，CUREC，工学院大学電子技術研究部，東京電機大学理工学部コンピュータ部，早稲田大学理工学部電気工学科Z80 CLUB，以上です。

**●コンパイラ時代到来か？**

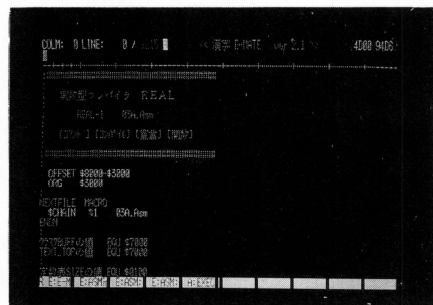
さて，S-OSにSmall-CとREALという強力な言語が２つも加わりました。しかし，どんな優秀な言語もアプリケーションが作られなくては宝の持ち腐れですね。

当然，大々的にアプリケーション，または周辺ソフトの募集です。C言語の移植性神話がどこまで通用するかも興味深いところです。特に今回Small-Cを配布した方々は鋭意投稿お願いいたします。

**●S-OSの系譜(22)**

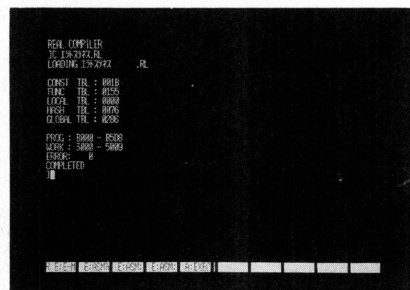
1987年８月号で，CPUの異なるFM-7にまでその増殖の手を伸ばしたS-OS"SWORD"は，続く９月号で再びNEC PCをターゲットにしました。PC-8801版はすでに1986年６月号で発表されていますが，このときのバージョンはPC-8801のROMを利用したものでした。動作速度の点などから，オールRAMバージョンの発表が待たれていたところです。同時に，PC-8001用のS-OS"SWORD"を望む声も編集室に寄せられていました。

８月号ではこれらの要望に応えるかたちで，PC-8001/8801用のオールRAM版"SWORD"が発表されました。オールRAM版の実現方法は少々凝ったものでした。まず，XBIOSというX1と互換性を持ったBIOSを作成し，その上でX1用のS-OS"SWORD"を走らせているのです。ハードウェアの違いと，XBIOSがX1のBIOSにフルコンパチではないことから，HuBASICなどのX1用のソフトウェアは当然実行できません。しかし，S-OSを1から作成する必要のないこと，発表後に発見されたバグをとった"SWORD"を使用できるため比較的安定していることなどのメリットがあります。

また８月号では，リロケータブル逆アセンブラInside-Rが発表されています。S-OSの標準的なデバッガであるZAIDは，ZEDAでのプログラム作成とそのデバッグを並行して行うことを考慮し，$5000_H$で動作するようになっています。そのため，解析しようとするプログラムのアドレスがZAIDと重なってしまってはどうしようもありません。Inside-RはS-OSに用意されたGETPC，［HL］の２つの機能を使い，どのアドレスでも動作するリロケータブルな逆アセンブラとなっていました。ZAIDからブレイクポイント設定機能を削除し，代わりにアドレスのクロスリファレンス作成機能などを装備してわずか2Kバイト。強力な逆アセンブラです。

全機種共通
S-OS"SWORD"要

# ソースリスト編
# REAL

*Ohnuki Nobuaki*
**大貫 信昭**

**お待たせしました。5月号で発表した実数型コンパイラ言語REALのソースリストを掲載します。存分に堪能してください。あまりに大きなプログラムのため掲載が遅れましたことをおわびします。**

## 実数型コンパイラREAL

REALはS-OSでは初めて実数型を直接サポートしたコンパイラ言語です。基本的な部分はこれまでS-OS上の構造化コンパイラ言語として定評のあったSLANGとほぼ同じ仕様となっており，1990年3月号で発表された浮動小数点演算パッケージSOROBANを利用して実数演算を実現したものです。

従来，8ビットCPU上で実数型コンパイラを作成することはかなりの困難を伴うものとして扱われていました。これはCP/M上の多くのコンパイラ言語や，これまで発表されてきたS-OS用の言語群が整数型仕様なのを見てもわかると思います。

もともとSOROBANはこういった状況を打開するために独立したパッケージとして発表されたものでした。それが発表以来1年2カ月を経て，ようやくこれを使用したコンパイラ言語の発表となったわけです。

今回はこのREALのソースリストをまとめて掲載します。実行ファイル自体はダンプリストとして1991年5月号に掲載されていますので，実際にプログラムを使用するときにはそちらを入力してください。

*　　＊　　＊

もちろん，6月号のごめんなさいのコーナーで発表されたバグは修正されていますので，5月号に載ったダンプリストとは一部異なる点もあります。気をつけてください。

## 入力方法

プログラムはいつもお馴染みのOHM-Z80で記述されています。S-OSの拡張仕様である漢字を多用していますので，このソースリストをそのまま打ち込んでアセンブルすることは不可能，ないしは非常に困難なことだと思われます。その分，無味乾燥なアセンブラソースリストよりもドキュメント性は格段に高いと思います。これからコンパイラ言語を作ろうという方は参考にするとよいかもしれません。

それでも入力するという方のために，一応アセンブルの手順を示しておきましょう。まず，アセンブラとして1990年3月号で発表されたOHM-Z80が必要です。それ以外のアセンブラではアセンブルできません。

リスト1のソースリストは4つのプログラムが連続して出力されています。それぞれの部分を，

```
REAL-1.ASM
REAL-2.ASM
REAL-3.ASM
REAL-4.ASM
```

というファイル名で分割して入力，セーブしてください。次に，

```
#L  OHM-Z80
#J3000
```

のようにOHM-Z80を起動し，

```
)A REAL-1.ASM
```

でアセンブルを開始します。あとは自動的に次のソースプログラムを読み込んでアセンブルを終了します。

こうしてできあがったオブジェクトはオフセットつきでアセンブルされたものですから，

```
)S REAL:3000:6FFF:3000:8000
```

のようにオフセット(この場合は8000H)を指定してセーブすると，そのまま起動できる実行ファイルができあがります。

S-OS変身セットでバッチ処理を追加している場合ならば，これらの操作をまとめて行うとよいでしょう。

なお，できあがったプログラムの使用方法，文法などについては5月号の解説をお読みください。

## Small-CとREAL

S-OSの主流言語はいうまでもなく「アセンブリ言語」です。高級言語の主流はSLANGでした。これからの主流となると思われるのはSLANG/REAL，Small-Cでしょう。

SLANGはS-OSの実情を踏まえて開発されたものであり，"SWORD"の機能をかなり使い切ったシステムといえるでしょう。

それに対してCはもともとミニコンとUNIXの環境で育った言語です。多くの魅力を持った言語ですが，現在の"SWORD"ではC言語をおいしく使用するための環境を提供できないという問題が浮かび上がってきます。かなりC言語の仕様を意識したSLANGがなぜC言語そのものではないのかというのもそういった点に起因するものと思われます。また，構造体の使えないC処理系では移植性がどの程度保証できるのかも定かではありません。

コンパイラ言語としてのREALとSmall-Cが棲み分けする余地は十分にあるといえます。またメジャーなC言語とS-OSを取り持つ環境を作ったり，新たなコンパイラを作ることも考えてみるべきでしょう。

## リスト1

```
0000                      1 ;##########################################
0000                      2 :
0000                      3 :        実数型コンパイラ  REAL
0000                      4 :
0000                      5 :          REAL-1    03A.Asm
0000                      6 :
0000                      7 :        [ｺﾏﾝﾄﾞ] [ｺﾝﾊﾟｲﾙ] [宣言] [関数]
0000                      8 :
0000                      9 ;##########################################
0000                     10 :
0000                     11        OFFSET $8000-$3000
3000                     12        ORG    $3000
3000                     13 :
3000                     14 NEXTFILE  MACRO
3000                     15        $CHAIN   %1    03A.Asm
3000                     16 ENDM
3000                     17 :
7000                     18 ｸﾗｽﾀBUFFの値    EQU $7000
7000                     19 TEXT_TOPの値    EQU $7000
3000                     20 :
0100                     21 定数表SIZEの値 EQU $0100
0200                     22 関数WKSIZEの値 EQU $0200
0100                     23 局所表SIZEの値 EQU $0100
0400                     24 ﾊｯｼｭ表SIZEの値 EQU $0400
3000                     25 :
0020                     26 名前最大長    EQU 32
0080                     27 LINEBUFF長    EQU 128
0008                     28 TEMP数        EQU 8
0008                     29 定数ｽﾀｯｸ数    EQU 8
0014                     30 式ｽﾀｯｸ数      EQU 20
0008                     31 引数最大数    EQU 8
0005                     32 M引数最大数 EQU 5  :HL,DE,BC,IX,IY
0010                     33 LOOP入子上限 EQU 16
0004                     34 INCL入子上限 EQU 4
3000                     35 :
3000                     36 :       S-OS CALL
3000                     37 :
1FFA                     38 #HOT    EQU $1FFA
1F8E                     39 #MON    EQU $1F8E
1F81                     40 [HL]    EQU $1F81
1F94                     41 #PEEK   EQU $1F94
1F91                     42 #PEEK@  EQU $1F91
1F9A                     43 #POKE   EQU $1F9A
1F97                     44 #POKE@  EQU $1F97
2030                     45 #WIDCH  EQU $2030
201B                     46 #SCRN   EQU $201B
201E                     47 #LOC    EQU $201E
2018                     48 #CSR    EQU $2018
3000                     49 :
1FB2                     50 #HLHEX  EQU $1FB2
1FB5                     51 #AHEX   EQU $1FB5
1FB8                     52 #HEX    EQU $1FB8
1FBB                     53 #ASC    EQU $1FBB
3000                     54 :
1FD3                     55 #GETL   EQU $1FD3
1FD0                     56 #GETKY  EQU $1FD0
1FCA                     57 #INKEY  EQU $1FCA
2021                     58 #FLGET  EQU $2021
1FC7                     59 #PAUSE  EQU $1FC7
3000                     60 :
1FC1                     61 #PRTHX  EQU $1FC1
1FBE                     62 #PRTHL  EQU $1FBE
1FC4                     63 #BELL   EQU $1FC4
1FE2                     64 #MPRNT  EQU $1FE2
1FE5                     65 #MSX    EQU $1FE5
1FE8                     66 #MSG    EQU $1FE8
1FEE                     67 #LTNL   EQU $1FEE
1FEB                     68 #NL     EQU $1FEB
1FF4                     69 #PRINT  EQU $1FF4
1FF1                     70 #PRNTS  EQU $1FF1
1FDC                     71 #LPRNT  EQU $1FDC
1FD9                     72 #LPTON  EQU $1FD9
1FD6                     73 #LPTOF  EQU $1FD6
3000                     74 :
2006                     75 #DIR    EQU $2006
2024                     76 #RDVSW  EQU $2024
2027                     77 #SDVSW  EQU $2027
1F9D                     78 #FPRNT  EQU $1F9D
1FA3                     79 #FILE   EQU $1FA3
2000                     80 #DRDSB  EQU $2000
2009                     81 #ROPEN  EQU $2009
1FA6                     82 #RDD    EQU $1FA6
1FAF                     83 #WOPEN  EQU $1FAF
1FAC                     84 #WRD    EQU $1FAC
2033                     85 #ERROR  EQU $2033
3000                     86 :
3000                     87 :       S-OS WORK
3000                     88 :
1F6A                     89 #MEMAX  EQU $1F6A
1F68                     90 #WKSIZ  EQU $1F68
3000                     91 :
1F5D                     92 #DSK    EQU $1F5D
1F5E                     93 #FATPOS EQU $1F5E
1F62                     94 #FATBF  EQU $1F62
1F64                     95 #DTBUF  EQU $1F54
3000                     96 :
1F70                     97 #DTADR  EQU $1F70
1F72                     98 #SIZE   EQU $1F72
1F6E                     99 #EXADR  EQU $1F6E
1F74                    100 #IBFAD  EQU $1F74
3000                    101 :
1F76                    102 #KBFAD  EQU $1F76
3000                    103 :
3000                    104 :       SOROBAN CALL
3000                    105 :
0000                    106 #PRCSN  EQU $00
0002                    107 #MOVE   EQU $02
000E                    108 #CVSTF  EQU $0E
0011                    109 #CVUTF  EQU $11
0014                    110 #CVITF  EQU $14
0017                    111 #CVFTS  EQU $17
001A                    112 #CVFTU  EQU $1A
001D                    113 #CVFTI  EQU $1D
0020                    114 #ADD    EQU $20
0023                    115 #SUB    EQU $23
0026                    116 #MUL    EQU $26
0029                    117 #DIV    EQU $29
002F                    118 #MOD    EQU $2F
0032                    119 #CMP    EQU $32
0035                    120 #NEG    EQU $35
0038                    121 #INT-   EQU $38
003B                    122 #FIX    EQU $3B
003E                    123 #FRAC   EQU $3E
0041                    124 #CINT   EQU $41
0044                    125 #SQR    EQU $44
0047                    126 #SIN    EQU $47
004A                    127 #COS    EQU $4A
004D                    128 #TAN    EQU $4D
0050                    129 #ATN    EQU $50
0053                    130 #EXP    EQU $53
0056                    131 #LOG    EQU $56
0059                    132 #POW    EQU $59
005F                    133 #RAD    EQU $5F
0062                    134 #ABS    EQU $62
0065                    135 #SGN    EQU $65
3000                    136 :
3000                    137 :       型
3000                    138 :
0001                    139 VOID$   EQU 1
0002                    140 CHR$    EQU 2
0003                    141 INT$    EQU 3
0004                    142 REAL$   EQU 4
3000                    143 :
3000                    144 :       表域
3000                    145 :
0000                    146 大域$   EQU 0
0001                    147 局所$   EQU 1
3000                    148 :
3000                    149 :       名前ﾁｪｯｸ
3000                    150 :
3000                    151 :ｴﾗｰ    EQU 0
0001                    152 名前@   EQU 1
0002                    153 関数@   EQU 2
0003                    154 配列@   EQU 3
3000                    155 :
3000                    156 :       式ｽﾀｯｸのﾏｰｸ
3000                    157 :
0000                    158 中間%   EQU 0
0001                    159 定数%   EQU 1
0002                    160 変数%   EQU 2
0003                    161 ｺｰﾄﾞ%   EQU 3
3000                    162 :
3000                    163 :       表に登録する型
3000                    164 :
3000                    165 :なし   EQU 0
0001                    166 記定$   EQU 1
0002                    167 変数$   EQU 2
0003                    168 関変1$  EQU 3
0004                    169 関変2$  EQU 4  :関変1$+1
0005                    170 配列1$  EQU 5
0006                    171 配列2$  EQU 6  :配列1$+1
0007                    172 ﾗﾍﾞﾙ$   EQU 7
0008                    173 未ﾗﾍﾞﾙ$ EQU 8
0009                    174 関数$   EQU 9
000A                    175 未関数$ EQU 10 ;関数$+1
000B                    176 M関$    EQU 11
000C                    177 未M関$  EQU 12 ;M関$+1
3000                    178 :
3000                    179 :       CONST
3000                    180 :
0001                    181 TRUE  EQU 1
0000                    182 FALSE EQU 0
0001                    183 ON    EQU TRUE
0000                    184 OFF   EQU FALSE
0001                    185 有    EQU TRUE
0000                    186 無    EQU FALSE
3000                    187 :
0011                    188 SKIP  EQU $11 :LD DE,nn
3000                    189 :
FFFF                    190 NULL  EQU -1
3000                    191 :
003E                    192 @A     EQU $3E
0021                    193 @HL    EQU $21
0011                    194 @DE    EQU $11
002A                    195 @HL[]  EQU $2A
5BED                    196 @DE[]  EQU $5BED
0022                    197 @[]HL  EQU $22
53ED                    198 @[]DE  EQU $53ED
0023                    199 +$     EQU $23
002B                    200 -$     EQU $2B
3000                    201 :
3000                    202 :       表の形式
3000                    203 :
3000                    204 :         "名前"
3000                    205 :         $0D
3000                    206 :         TYPE       (1)  ;例) 関数$
3000                    207 :         型         (1)  ;例) REAL$
3000                    208 :         表ﾃﾞｰﾀ     (2)  ;例) 関数のｱﾄﾞﾚｽ
3000                    209 :         補助ﾃﾞｰﾀ   (2)  ;例) 関数WKﾎﾟｲﾝﾀ
3000                    210 :         $00
3000                    211 :
3000                    212 :       関数WK
3000                    213 :         引数ｱﾄﾞﾚｽ  (2)    ﾁｪｲﾝ
3000                    214 :         引数の数   (1)
3000                    215 :         第1引数の型(1)
3000                    216 :         第2引数の型(1)
3000                    217 :           :
3000                    218 :           :
3000                    219 :         第8引数の型(1)
3000                    220 :
3000                    221 ;##########################################
3000                    222 :
3000 C3 17 30           223        JP  [ｺｰﾙﾄﾞｽﾀｰﾄ]
3003 18 5E              224        JR  [ﾎｯﾄｽﾀｰﾄ]
3005                    225 :
3005                    226 ;##########################################
3005                    227 :
3005                    228 :       変更可能な初期値
3005                    229 :
3005 01                 230 DISK          DB TRUE
3006 00 B0              231 OBJ初期値     DW $B000
3008 82 6E              232 ﾗﾝﾀｲﾑ最終ADR DW &RUNTIME_END-1
300A                    233 :
300A 00 70              234 ｸﾗｽﾀBUFF      DW ｸﾗｽﾀBUFFの値
300C 00 70              235 TEXT_TOP      DW TEXT_TOPの値
300E                    236 :
300E 00 01              237 定数表SIZE    DW 定数表SIZEの値
3010 00 02              238 関数WKSIZE    DW 関数WKSIZEの値
3012 00 01              239 局所表SIZE    DW 局所表SIZEの値
3014 00 04              240 ﾊｯｼｭ表SIZE    DW ﾊｯｼｭ表SIZEの値
3016                    241 :
3016 01                 242 BAT対応       DB TRUE
3017                    243 :
3017                    244 ;##########################################
3017                    245 :
3017                    246 [ｺｰﾙﾄﾞｽﾀｰﾄ]
3017 21 00 00 22 8A 45  247        (定数表TOP)=HL=0
301D ED 5B 0E 30 19     248        DE=(定数表SIZE) ADD HL,DE
3022 22 8C 45           249        (定数表END)=HL
3025 22 14 38           250        (関数WKTOP)=HL
3028 ED 5B 10 30 19     251        DE=(関数WKSIZE) ADD HL,DE
302D 22 16 38           252        (関数WKEND)=HL
3030 22 61 5A           253        (局所表TOP)=HL
3033 ED 5B 12 30 19     254        DE=(局所表SIZE) ADD HL,DE
3038 22 65 5A           255        (ﾊｯｼｭ表TOP)=HL
303B ED 5B 14 30 19     256        DE=(ﾊｯｼｭ表SIZE) ADD HL,DE
3040 22 67 5A           257        (ﾊｯｼｭ表END)=HL
3043 22 5D 5A           258        (大域表TOP)=HL
3046                    259 :
3046 11 28 00           260        DE=名前最大長+8
3049 2A 68 1F B7 ED 52  261        HL=(#WKSIZ)     SUB HL,DE
304F 22 5F 5A           262        (大域表END)=HL
3052 2A 65 5A B7 ED 52  263        HL=(ﾊｯｼｭ表TOP)  SUB HL,DE
3058 22 63 5A           264        (局所表END)=HL
305B                    265 :
305B CD 6B 58           266        [大域表初期化]
305E                    267 :
305E CD 1C 63 0C 00     268        CALL @MPRNT DM $0C,0
3063                    269 :
3063                    270 [ﾎｯﾄｽﾀｰﾄ]
3063 CD 1C 63 FB 20 43 4F  271     CALL @MPRNT DM @REAL," COMPILER",$0D,0
306A 4D 50 49 4C 45 52 0D
3071 00
3072                    272 :
3072 CD C2 69 C8        273        &[SRBCHECK] IF Z RET
3076                    274 :      HL=($1F82) IF HL=0 RET
3076                    275 :      INC HL/HL
3076                    276 :      E=(HL) INC HL
3076                    277 :      D=(HL)     IF DE=0 RET
3076 ED 53 CC 33        278        (SOROBANｱﾄﾞﾚｽ)=DE
307A                    279 :
307A ED 73 96 30        280        (HOTｽﾀｯｸ)=SP
307E                    281 [HOT]
307E                    282        {
307E ED 7B 96 30        283          SP=(HOTｽﾀｯｸ)
3082 CD EB 1F           284          CALL #NL
3085 3E 5D CD F4 1F     285          A="]"        CALL #PRINT
308A ED 5B 76 1F CD D3 1F  286       DE=(#KBFAD) CALL #GETL
3091 CD 9F 30           287          [CMD]
3094 18 E8              288        }
3096                    289 :
3096 00 00              290 HOTｽﾀｯｸ DW $0000
3098                    291 :
3098                    292 :
3098                    293 [SN_ERR]
3098 3E 0D              294        A=13
309A                    295 [ABORT]
309A CD 33 20 18 DF     296        CALL #ERROR  JR [HOT]
309F                    297 :
309F                    298 :
309F                    299 [CMD]
309F 1A FE 5D C0        300        IF (DE)<>"]" RET
30A3 13                 301        INC DE
30A4 1A CD DD 60 13     302        A=(DE) [TOUPPER]  INC DE
30A9                    303 :
30A9                    304 :      IF A=":"             RET
30A9 FE 21 CA FA 1F     305        IF A="!" CALL #HOT  RET
30AE FE 4D CA 8E 1F     306        IF A="M" CALL #MON  RET
30B3                    307 :
30B3 FE 43 CA 9C 31     308        IF A="C" [ｺﾝﾊﾟｲﾙ]   RET
30B8 FE 4F CA 99 5A     309        IF A="O" [CMD_LIST] RET
30BD FE 44 CA D2 30     310        IF A="D" [CMD_DIR]  RET
30C2 FE 4A CA 03 31     311        IF A="J" [CMD_JUMP] RET
30C7 FE 53 CA 1E 31     312        IF A="S" [CMD_SAVE] RET
30CC FE 58 CA 0D 31     313        IF A="X" [CMD_EXCG] RET
30D1 C9                 314        RET
30D2                    315 :
30D2                    316 :
30D2                    317 [CMD_DIR]
30D2 1A CD DD 60        318        A=(DE) [TOUPPER]
30D6                    319 :DV
30D6 FE 56 20 09        320        IF A="V" THEN
30DA 13                 321          INC DE
30DB CD ED 30 CD 27 20  322          [GETDEV]  CALL #SDVSW
30E1                    323 :D
30E1 18 09              324        ELSE
30E3 CD ED 30 CD 06 20 DC  325       [GETDEV]  CALL #DIR  IF C [ABORT]
30EA 9A 30
30EC                    326        FI
30EC C9                 327        RET
30ED                    328 :
30ED                    329 :
30ED                    330 [GETDEV]
30ED CD 93 31 13 1A 18  331        [SPCUT_DE]  INC DE  A=(DE)  DEC DE
30F3                    332 :
30F3 FE 3A 20 05 1A 13 13  333     IF A=":" : A=(DE)  INC DE/DE
30FA 18 03 CD 24 20     334        ELSE     : CALL #RDVSW
30FF                    335        FI
30FF 32 5D 1F C9        336        (#DSK)=A  RET
3103                    337 :
3103                    338 :
3103                    339 [CMD_JUMP]
3103 CD 93 31           340        [SPCUT_DE]
3106 CD B2 1F D4 81 1F  341        CALL #HLHEX  IF NC CALL [HL]
310C C9                 342        RET
310D                    343 :
310D                    344 :
310D                    345 [CMD_EXCG]
310D CD 93 31           346        [SPCUT_DE]
3110 CD B2 1F 38 03 22 0C  347     CALL #HLHEX  IF NC THEN (TEXT_TOP)=HL
3117 30
3118                    348 :
3118 2A 0C 30 C3 BE 1F  349        HL=(TEXT_TOP)  CALL #PRTHL  RET
311E                    350 :
311E                    351 :
311E                    352 [CMD_SAVE]
311E 21 98 30 E5        353        HL=[SN_ERR]  PUSH HL
3122                    354 :
3122 3E 01 CD A3 1F     355        A=1 CALL #FILE
3127                    356 :先頭
3127 1A FE 3A C0        357        A=(DE)  IF A<>":" RET
312B 13                 358        INC DE
312C CD B2 1F D8        359        CALL #HLHEX  IF C RET
3130 22 70 1F           360        (#DTADR)=HL
3133 22 6E 1F           361        (#EXADR)=HL
3136 22 91 31           362        (OFDATA)=HL
3139                    363 :最終
3139 1A FE 3A C0        364        A=(DE)  IF A<>":" RET
313D 13                 365        INC DE
313E CD B2 1F D8        366        CALL #HLHEX  IF C RET
3142                    367 :      RCF
3142 ED 4B 70 1F        368        BC=(#DTADR)
3146 ED 42              369        SBC HL,BC
3148 23                 370        INC HL
3149 22 72 1F           371        (#SIZE)=HL
314C                    372 :実行
314C 1A                 373        A=(DE)
314D FE 3A 20 15        374        IF A=":" THEN
3151 13                 375          INC DE
3152 CD B2 1F D8        376          CALL #HLHEX  IF C RET
3156 22 6E 1F           377          (#EXADR)=HL
3159                    378 :格納
3159 1A                 379          A=(DE)
315A FE 3A 20 08        380          IF A=":" THEN
315E 13                 381            INC DE
315F CD B2 1F D8        382            CALL #HLHEX  IF C RET
3163 22 91 31           383            (OFDATA)=HL
3166                    384          FI
3166                    385        FI
3166                    386 :
3166 CD 1C 63 57 52 49 54  387     CALL @MPRNT DM "WRITING ",0
316D 54 49 4E 47 20 00
3173 CD 9D 1F           388        CALL #FPRNT
3176 CD AF 1F DC 9A 30  389        CALL #WOPEN IF C [ABORT]
317C                    390 :
317C 2A 91 31 22 70 1F  391        (#DTADR)=HL=(OFDATA)
3182                    392 :
3182 CD AC 1F DC 9A 30  393        CALL #WRD   IF C [ABORT]
3188 CD 1C 63 0D FA 0D 00  394     CALL @MPRNT DM $0D,@COMPLETED,$0D,0
318F                    395 :
318F E1                 396        POP HL
3190 C9                 397        RET
3191                    398 :
3191 00 00              399 OFDATA  DW $0000
3193                    400 :
3193                    401 :
3193                    402 [SPCUT_DE]
3193 1A FE 20 20 03 13 18  403     UNTIL (DE)<>" " { INC DE }
319A F8
319B C9                 404        RET
319C                    405 :
319C                    406 ;##########################################
319C                    407 :
319C                    408 [ｺﾝﾊﾟｲﾙ]
319C 3E 01 32 F7 5C 3E 00  409     (#ｺﾝﾊﾟｲﾙ)=TRUE  (#IFﾓｰﾄﾞ)=FALSE
31A3 32 F8 5C
31A6                    410 :
31A6 AF 32 98 5D        411        (INCL入子)=0
31AA                    412 :
31AA 3E 00 32 06 5F     413        (ﾘｽﾄ表示SW)=OFF
31AF 1A FE 2F 20 06     414        IF (DE)="/" THEN
31B4 13 3E 01 32 06 5F  415          INC DE  (ﾘｽﾄ表示SW)=ON
31BA                    416        FI
31BA                    417 :
31BA 1A B7 20 19        418        IF (DE)=0 THEN
31BE 3E 01 32 38 33     419          (ONﾒﾓﾘ)=TRUE
31C3 2A 0C 30 22 70 1F  420          (#DTADR)=HL=(TEXT_TOP)
31C9 44 4D 7E 23 B7 20 FB  421       BC=HL  { A=(HL)  INC HL } UNTIL A=0
31D0 ED 42              422          SBC HL,BC
31D2 22 72 1F           423          (#SIZE)=HL
31D5                    424 :
31D5 18 19              425        ELSE
31D7 3A 05 30 FE 01 20 0A  426       IF (DISK)=TRUE THEN
31DE 3E 00 32 38 33 CD 1B  427         (ONﾒﾓﾘ)=FALSE  [OPEN]
31E5 5F
31E6 18 08              428          ELSE
31E8 3E 01 32 38 33 CD 5D  429         (ONﾒﾓﾘ)=TRUE   [LOAD]
31EF 5F
31F0                    430          FI
31F0                    431        FI
31F0                    432 :
31F0 AF 32 08 64        433        (ｴﾗｰ回数)=0
31F4                    434 :
31F4 3E 01 32 E2 39     435        (未ｺｰﾙ回数)=1 ;for CALL MAIN()
31F9                    436 :
31F9 CD 6B 58           437        [大域表初期化]
31FC CD 8A 58           438        [局所表初期化]
31FF                    439 :
31FF                    440 :定数表初期化
31FF 2A 8A 45 22 8E 45  441        (定数表ﾎﾟｲﾝﾀ)=HL=(定数表TOP)
3205 AF CD 9A 1F        442        A=0  CALL #POKE
3209                    443 :
3209                    444 :関数WK初期化
3209 2A 14 38 22 18 38  445        (関数WKﾎﾟｲﾝﾀ)=HL=(関数WKTOP)
320F                    446 :
320F CD C8 4F           447        [TEMPﾁｪｲﾝ初期化]
3212                    448 :
3212 CD 12 5E           449        [TEXT初期化]
3215                    450 :
3215 CD E1 64           451        [ﾗﾝﾀｲﾑSRB設定]
3218                    452 :
3218 21 00 00           453        HL=0
321B 22 58 59           454        (ﾊｯｼｭ表登録数)=HL
321E 22 77 34           455        (局所表最尾ﾎﾟｲﾝﾀ)=HL
3221                    456 :
3221 CD 5C 33 CD 09 64 CD  457     [ｱﾄﾞﾝｽ宣言] [組込登録] [ﾗﾝﾀｲﾑ転送]
3228 F3 64
322A                    458 :
322A CD E9 5D           459        [ｱﾄﾞﾝｽ_ﾁｪｯｸ]
322D                    460        {
322D CD 79 34           461          [大域宣言]
3230 CD 20 34           462          [ブロック]
3233 CD AC 5B           463          [SPCUT]
3236 B7 20 F4           464        } UNTIL A=0
3239                    465 :
3239 CD 6A 45           466        [定数ﾁｪｲﾝ]
323C                    467 :
323C 3A F8 5C FE 01 20 03  468     IF (#IFﾓｰﾄﾞ)=TRUE THEN
3243 CD A2 62 E4 23 45 4E  469       [ERROR2] DM @MIS,"#ENDIF",0
324A 44 49 46 00
324E                    470        FI
324E                    471 :
324E 3A E2 39 B7 28 0E  472        IF (未ｺｰﾙ回数)<>0 THEN
3254 CD A2 62 EC F0 00  473          [ERROR2] DM @UDEF,@FUNC,0
325A                    474 :
325A 3E 00 32 51 5A CD 62  475       (表域)=大域$  [未宣言LIST]
3261 5A
3262                    476        FI
3262                    477 :
3262 CD EE 1F           478        CALL #LTNL
3265                    479 :
3265 CD 1C 63 EF 00     480        CALL @MPRNT DM @CONST,0
326A 2A 8E 45           481        HL=(定数表ﾎﾟｲﾝﾀ)
326D ED 5B 8A 45 CD 3C 33  482     DE=(定数表TOP)  CALL SUBPRT
3274                    483 :
3274 CD 1C 63 F0 20 00  484        CALL @MPRNT DM @FUNC," ",0
327A 2A 18 38           485        HL=(関数WKﾎﾟｲﾝﾀ)
327D ED 5B 14 38 CD 3C 33  486     DE=(関数WKTOP)  CALL SUBPRT
3284                    487 :
3284 CD 1C 63 F8 20 00  488        CALL @MPRNT DM @LOCAL," ",0
328A 2A 77 34           489        HL=(局所表最尾ﾎﾟｲﾝﾀ)
328D ED 5B 61 5A CD 3C 33  490     DE=(局所表TOP)  CALL SUBPRT
3294                    491 :
3294 CD 1C 63 F9 20 20 00  492     CALL @MPRNT DM @HASH,"  ",0
329B 2A 58 59           493        HL=(ﾊｯｼｭ表登録数)
329E 29 11 00 00 CD 3C 33  494     ADD HL,HL DE=0  CALL SUBPRT
32A5                    495 :
```

```
32A5 CD 1C 63 F6 00         496         CALL @MPRNT DM @GLBL,0
32AA 2A 59 5A               497         HL=(大域表ﾎﾟｲﾝﾀ)
32AD ED 5B 5D 5A CD 3C 33   498         DE=(大域表TOP)  CALL SUBPRT
32B4                        499 ;Program
32B4 CD EE 1F               500         CALL #LTNL
32B7 CD 1C 63 50 52 4F 47   501         CALL @MPRNT  DM "PROG : ",0
32BE 20 3A 20 00
32C2 2A C3 33 FD E5 D1 1B   502         HL=(OBJ先頭) DE=IY DEC DE  PRT[HLDE]
32C9 CD 4A 33
32CC                        503 ;Object
32CC ED 4B C5 33            504         BC=(OFFSET)
32D0 78 B1 28 19            505         IF BC<>0 THEN
32D4 CD 1C 63 4F 42 4A 20   506           CALL @MPRNT DM "OBJ  : ",0
32DB 20 3A 20 00
32DF 2A C3 33 09            507           HL=(OBJ先頭)  ADD HL,BC
32E3 FD E5 D1 1B EB 09 EB   508           DE=IY DEC DE  ADD DE,BC  PRT[HLDE]
32EA CD 4A 33
32ED                        509         FI
32ED                        510 ;Work
32ED 3A C7 33 FE 01 20 15   511         IF (WORK指定)=TRUE THEN
32F4 CD 1C 63 57 4F 52 4B   512           CALL @MPRNT DM "WORK : ",0
32FB 20 3A 20 00
32FF CD 7E 36               513           [GETWORKﾎﾟｲﾝﾀ]
3302 EB 2A C8 33 CD 4A 33   514           EX DE,HL  HL=(WORK先頭)  PRT[HLDE]
3309                        515         FI
3309                        516 ;
3309 3A FA 33 FE 01 20 12   517         IF (STACK指定)=TRUE THEN
3310 ED 5B C3 33            518           DE=(OBJ先頭)
3314 FD 21 10 00 FD 19      519           IY=&STACK-&R  ADD IY,DE
331A 3E 31 2A FB 33 CD 73   520           A=$31  HL=(STACK)  @[IY]AHL
3321 5B
3322                        521         FI
3322                        522 ;
3322 CD 1C 63 E0 3A 00      523         CALL @MPRNT DM @ERR,":",0
3328 26 00                  524         H=0
332A 3A 08 64 6F CD 0C 5F   525         L=(ｴﾗｰ回数) [PRT10]
3331 CD 1C 63 0D FA 0D 00   526         CALL @MPRNT DM $D,@COMPLETED,$D,0
3338 C3 C4 1F               527         CALL #BELL  RET
333B                        528 :
333B 00                     529 ONﾒﾓﾘ DB 0
333C                        530 ;
333C                        531 ;
333C                        532 SUBPRT
333C 37 ED 52               533         SUB HL,DE
333F E5                     534         PUSH HL
3340 CD 1C 63 ED 3A 20 00   535           CALL @MPRNT DM @TBL,": ",0
3347 E1                     536         POP  HL
3348 18 0C                  537         JR PRT[HLDE]1
334A                        538 ;
334A                        539 PRT[HLDE]
334A D5                     540         PUSH DE
334B CD BE 1F               541         CALL #PRTHL
334E CD E2 1F 20 2D 20 00   542         CALL #MPRNT  DM " - ",0
3355 E1                     543         POP  HL
3356                        544 PRT[HLDE]1
3356 CD BE 1F               545         CALL #PRTHL
3359 C3 EE 1F               546         CALL #LTNL  RET
335C                        547 ;
335C                        548 ;
335C                        549 [ｱﾄﾞﾚｽ宣言]
335C CD 00 34               550         [DOUBLE宣言]              ;倍精度
335F                        551 ;
335F 2A 06 30 22 C3 33 E5   552         HL=(OBJ初期値)  (OBJ先頭)=HL  IY=HL
3366 FD E1
3368 21 00 00 22 C5 33      553         (OFFSET)=HL=0
336E 3E 00                  554         A=FALSE
3370 32 C7 33               555         (WORK指定)=A
3373 32 FA 33               556         (STACK指定)=A
3376                        557         {
3376 CD E3 39               558           [式初期化]
3379 CD 67 60               559           [TBLｻｰﾁ]
337C 4F 52 C7 CE 33         560           DM "ORG"+$80    DW [ORG宣言]
3381 57 4F 52 CB DF 33      561           DM "WORK"+$80   DW [WORK宣言]
3387 4F 46 46 53 45 D4 D8   562           DM "OFFSET"+$80 DW [OFFSET宣言]
338E 33
338F 53 54 41 43 CB EE 33   563           DM "STACK"+$80  DW [STACK宣言]
3396 53 49 4E 47 4C C5 FD   564           DM "SINGLE"+$80 DW [SINGLE宣言]
339D 33
339E 44 4F 55 42 4C C5 00   565           DM "DOUBLE"+$80 DW [DOUBLE宣言]
33A5 34
33A6 00                     566           DM 0
33A7 30 08                  567           IF NC EXIT
33A9                        568 ;
33A9 CD 81 1F CD FE 5F      569           CALL [HL]  [ｾﾐｺﾛﾝ]
33AF 18 C5                  570         }
33B1 2A C3 33               571         HL=(OBJ先頭)
33B4 ED 5B C5 33            572         DE=(OFFSET)
33B8 19 7D D6 00 7C DE 70   573         ADD HL,DE  IF HL<$7000 [不正ｱﾄﾞﾚｽ!]
33BF DC 51 61
33C2 C9                     574         RET
33C3                        575 ;
33C3 00 00                  576 OBJ先頭      DW $0000
33C5 00 00                  577 OFFSET       DW $0000
33C7 00                     578 WORK指定     DB 0
33C8 00 00                  579 WORK先頭     DW $0000
33CA 00 00                  580 WORKﾎﾟｲﾝﾀ    DW $0000
33CC 00 00                  581 SOROBANｱﾄﾞﾚｽ DW $0000
33CE                        582 ;
33CE                        583 [ORG宣言]
33CE CD 0A 34 22 C3 33 E5   584         [INT定数式!3000] (OBJ先頭)=HL IY=HL
33D5 FD E1
33D7 C9                     585         RET
33D8                        586 ;
33D8                        587 [OFFSET宣言]
33D8 CD 17 34 22 C5 33 C9   588         [INT定数式!] (OFFSET)=HL RET
33DF                        589 ;
33DF                        590 [WORK宣言]
33DF 3E 01 32 C7 33         591         (WORK指定)=TRUE
33E4 CD 0A 34               592         [INT定数式!3000]
33E7 22 C8 33               593         (WORK先頭)=HL
33EA 22 CA 33 C9            594         (WORKﾎﾟｲﾝﾀ)=HL  RET
33EE                        595 ;
33EE                        596 [STACK宣言]
33EE 3E 01 32 FA 33         597         (STACK指定)=TRUE
33F3 CD 0A 34 22 FB 33 C9   598         [INT定数式!3000] (STACK)=HL RET
33FA                        599 ;
33FA 00                     600 STACK指定 DB 0
33FB 00 00                  601 STACK     DW $0000
33FD                        602 ;
33FD 3E 05 11               603 [SINGLE宣言]  A=5  DB SKIP
3400 3E 08                  604 [DOUBLE宣言]  A=8
3402 2A CC 33 77 32 AE 59   605         HL=(SOROBANｱﾄﾞﾚｽ)  (HL)=A  (精度)=A
3409 C9                     606         RET
340A                        607 ;
340A                        608 [INT定数式!3000]
340A CD 17 34 7D D6 00 7C   609         [INT定数式!] IF HL<$3000 [不正ｱﾄﾞﾚｽ!]
3411 DE 30 DC 51 61
3416 C9                     610         RET
3417                        611 ;
3417                        612 [INT定数式!]
3417 CD 22 3A FE 01 C2 7E   613         [INT定数式]  IF A<>定数% JP [HOT]
341E 30
341F C9                     614         RET
3420                        615 ;
3420                        616 ;
3420                        617 [ブロック]
3420 CD 8A 58 CD C8 4F      618         [局所表初期化] [TEMPﾁｪｲﾝ初期化]
3426                        619 ;
3426 CD BD 37               620         [関数頭書き]
3429                        621 ;
3429 CD 7C 34               622         [局所宣言]
342C                        623 ;
342C CD A7 39               624         [関数LINK]
342F                        625 ;
342F AF 32 F0 55 32 4E 55   626         A=0  (LOOPﾚﾍﾞﾙ)=A  (未ﾗﾍﾞﾙ回数)=A
3436                        627 ;
3436 CD C6 51 F5            628         [BROCK前ｶｯｺ] PUSH AF
343A                        629         {
343A CD FD 51 38 05         630           [後ｶｯｺ] IF C EXIT
343F CD 4E 51               631           [文]
3442 18 F6                  632         }
3444 F1 CD B6 51            633         POP AF [後括弧ｴﾗｰ?]
3448                        634 ;
3448 CD 3F 5B C9            635         @[IY]SP1 : RET
344C                        636 ;
344C 3A 4E 56 B7 28 12      637         IF (未ﾗﾍﾞﾙ回数)<>0 THEN
3452 3E 01 32 51 5A         638           (表域)=局所$
3457 CD 6B 5A               639           [未宣言LIST]
345A CD 99 62 EC 4C 41 42   640           [ERROR] DM @UDEF,"LABEL",0
3461 45 4C 00
3464                        641         FI
3464                        642 ;
3464 2A 5B 5A               643         HL=(局所表ﾎﾟｲﾝﾀ)
3467 ED 5B 77 34            644         DE=(局所表最尾ﾎﾟｲﾝﾀ)
346B 7B 95 7A 9C 30 03 22   645         IF HL>DE THEN (局所表最尾ﾎﾟｲﾝﾀ)=HL
3472 77 34
3474                        646 ;
3474 C3 21 50               647         [TEMPﾁｪｲﾝ] RET
3477                        648 ;
3477 00 00                  649 局所表最尾ﾎﾟｲﾝﾀ DW $0000
3479                        650 ;
3479                        651 ;*************************************
3479                        652 ;
3479 3E 00 11               653 [大域宣言] A=大域$ DB SKIP
347C 3E 01                  654 [局所宣言] A=局所$
347E                        655 ;in A:大域$ :: 局所$
347E 32 51 5A               656         (表域)=A
3481                        657         {
3481 CD E3 39               658           [式初期化]
3484 CD 67 60               659           [TBLｻｰﾁ]
3487 43 4F 4E 53 D4 C2 34   660           DM "CONST"+$80   DW [記号定数宣言]
348E 56 41 D2 F6 34         661           DM "VAR"+$80     DW [変数宣言]
3493 41 52 52 41 D9 83 35   662           DM "ARRAY"+$80   DW [配列宣言]
349A 00                     663           DM 0
349B 38 1D                  664           IF NC THEN
349D 3A 51 5A FE 01 C8      665             IF (表域)=局所$ RET
34A3                        666 ;
34A3 CD 67 60               667             [TBLｻｰﾁ]
34A6 44 45 43 4C 41 52 C5   668             DM "DECLARE"+$80 DW [ﾌﾟﾛﾄﾀｲﾌﾟ宣言]
34AD B9 36
34AF 4D 41 43 48 49 4E C5   669             DM "MACHINE"+$80 DW [MACHINE宣言]
34B6 2C 37
34B8 00                     670             DM 0
34B9 D0                     671             IF NC RET
34BA                        672           FI
34BA                        673 ;
34BA CD 81 1F CD FE 5F      674           CALL [HL]  [ｾﾐｺﾛﾝ]
34C0 18 BF                  675         }
34C2                        676 ;
34C2                        677 ;
34C2                        678 [記号定数宣言]
34C2 3E 03 32 52 5A         679         (登録型)=INT$
34C7                        680         {
34C7 CD 86 37 30 03 32 52   681           [型ｻｰﾁ] IF C THEN (登録型)=A
34CE 5A
34CF CD AC 5B               682           [SPCUT]
34D2 3E 01 FD E5 E1 11 00   683           A=記定$  HL=IY  DE=0  [名前登録]
34D9 00 CD A7 36
34DD                        684 ;
34DD CD 1D 60 3D            685           [SPINC] DB "="
34E1                        686 ;
34E1 3A 52 5A FE 03 20 05   687           IF (登録型)=INT$ THEN
34E8 CD F9 59               688             [定数式表変更]
34EB                        689
34EB 18 03                  690           ELSE
34ED                        691 :           HL=IY  A=記定$ [表変更]
34ED CD F5 39               692             [CODE_REAL]
34F0                        693           FI
34F0                        694 ;
34F0 CD B7 37               695           [ｶﾝﾏ]
34F3 38 D2                  696         } UNTIL NC
34F5 C9                     697         RET
34F6                        698 ;
34F6                        699 ;
34F6                        700 [変数宣言]
34F6 3E 03 32 52 5A         701         (登録型)=INT$
34FB                        702         {
34FB CD 7E 36 CD 43 35      703           [GETWORKﾎﾟｲﾝﾀ]  [変数登録]
3501                        704 ;
3501 CD AC 5B               705           [SPCUT]
3504                        706 ::ｱﾄﾞﾚｽ
3504 FE 3A 20 07            707           IF A=":" THEN
3508 DD 23 CD F9 59         708             INC IX [定数式表変更]
350D                        709 :=初期値
350D 18 2E FE 3D 20 19      710           EF A="=" THEN
3513 DD 23                  711             INC IX
3515 FD E5 E1 CD FC 59      712             HL=IY [表DATA変更]
351B                        713 ;
351B 3A 52 5A FE 03 20 05   714             IF (登録型)=INT$ : [CODE_INT]
3522 CD EF 39
3525 18 03 CD F5 39         715             ELSE             : [CODE_REAL]
352A                        716             FI
352A                        717 ;
352A 18 11                  718           ELSE
352C 21 02 00               719             HL=2
352F 3A 52 5A FE 04 20 04   720             IF (登録型)=REAL$ THEN L=(精度)
3536 3A AE 59 6F
353A CD 8C 36               721             [ADDWORKﾎﾟｲﾝﾀ]
353D                        722           FI
353D                        723 ;
353D CD B7 37               724           [ｶﾝﾏ]
3540 38 B9                  725         } UNTIL NC
3542 C9                     726         RET
3543                        727 ;
3543                        728 ;
3543                        729 [変数登録] :in HL:登録data
3543 E5                     730         PUSH HL
3544                        731 ;
3544 CD 97 37 30 03 32 52   732         [型ｻｰﾁ2] IF C THEN (登録型)=A
354B 5A
354C                        733 ;
354C CD AC 5B CD 25 3D      734         [SPCUT] [名前ﾁｪｯｸ]
3552                        735 ;
3552 E1                     736         POP HL
3553                        737 ;
3553 FE 01 20 18            738         IF A=名前0 THEN
3557 3A 52 5A FE 02 20 0A   739           IF (登録型)=CHR$ THEN
355E E5                     740             PUSH HL
355F CD 04 62 3E 03 32 52   741             [型ｴﾗｰ]  (登録型)=INT$
3566 5A
3567 E1                     742             POP  HL
3568                        743           FI
3568 3E 02 11 00 00 CD A7   744           A=変数$  DE=0 [名前登録]
356F 36
3570                        745 ;
3570 18 10                  746         ELSE
3572 3E 03 11 00 00 CD A7   747           A=関変1$ DE=0 [名前登録]  [添字]
3579 36 CD D3 35
357D                        748 ;
357D 3E 03 32 52 5A         749           (登録型)=INT$
3582                        750         FI
3582 C9                     751         RET
3583                        752 ;
3583                        753 ;
3583                        754 [配列宣言]
3583 3E 03 32 52 5A         755         (登録型)=INT$
3588                        756         {
3588 CD 97 37 30 03 32 52   757           [型ｻｰﾁ2] IF C THEN (登録型)=A
358F 5A
3590 CD AC 5B               758           [SPCUT]
3593 CD 7E 36 3E 05 11 00   759           [GETWORKﾎﾟｲﾝﾀ]  A=配列1$  DE=0
359A 00
359B CD A7 36               760           [名前登録]
359E                        761 ;
359E CD D3 35               762           [添字]
35A1                        763 ;
35A1 CD AC 5B               764           [SPCUT]
35A4                        765 ::ｱﾄﾞﾚｽ
35A4 FE 3A 20 07            766           IF A=":" THEN
35A8 DD 23 CD F9 59         767             INC IX [定数式表変更]
35AD                        768 :=初期値
35AD 18 1E FE 3D 20 07      769           EF A="=" THEN
35B3 DD 23 CD 3D 36         770             INC IX [配列初期値]
35B8                        771 ;
35B8 18 13                  772           ELSE
35BA 2A 25 36 7C B5 20 03   773             HL=(添字DATA) IF HL=0 THEN HL=1
35C1 21 01 00
35C4 3A 52 5A CD 27 36      774             A=(登録型) [型倍]
35CA CD 8C 36               775             [ADDWORKﾎﾟｲﾝﾀ]
35CD                        776           FI
35CD                        777 ;
35CD CD B7 37               778           [ｶﾝﾏ]
35D0 38 B6                  779         } UNTIL NC
35D2 C9                     780         RET
35D3                        781 ;
35D3                        782 ;
35D3                        783 [添字]
35D3                        784 ;第1添字
35D3 CD 01 36 DD 7E 00 FE   785         [添字処理] IF (IX)<>"[" RET
35DA 5B C0
35DC                        786 ;
35DC                        787 ;第2添字
35DC 2A 25 36               788         HL=(添字DATA)
35DF E5                     789         PUSH HL
35E0                        790 ;
35E0 CD 01 36               791         [添字処理]
35E3                        792 ;
35E3 2A 57 5A ED 5B 25 36   793         HL=(表ﾎﾟｲﾝﾀ)  DE=(添字DATA)
35EA                        794 ;
35EA CD 03 5A               795         [DEC_POKE_DE]           ;補助DATA
35ED                        796 ;       DEC HL  A=D     #POKE ;補助DATA
35ED                        797 ;       DEC HL  A=E     #POKE ;補助DATA
35ED 2B                     798         DEC HL                ;表DATA
35EE 2B                     799         DEC HL                ;表DATA
35EF 2B                     800         DEC HL                ;型
35F0 2B CD 94 1F            801         DEC HL          #PEEK ;TYPE
35F4 C6 01                  802         ADD A,関変2$-関変1$
35F6                        803 ;       ADD A,配列2$-配列1$
35F6 CD 9A 1F               804                         #POKE
35F9                        805 ;       DE=(添字DATA)
35F9 E1                     806         POP HL
35FA CD 23 6A 22 25 36 C9   807         &[乗算]  (添字DATA)=HL  RET
3601                        808 ;
3601                        809 ;
3601                        810 [添字処理]
3601 CD 1D 60 5B            811         [SPINC] DB "["
3605                        812 ;
3605 CD AC 5B 21 00 00      813         [SPCUT] HL=0
360B FE 5D 28 0E            814         IF A<>"]" THEN
360F CD 22 3A               815           [INT定数式]
3612 7C FE 80 D4 71 62      816           IF H>=$80 [範囲OVER]
3618 7C B5 CC 71 62         817           IF HL=0   [範囲OVER]
361D                        818         FI
361D                        819 ;
361D 22 25 36               820         (添字DATA)=HL
3620                        821 ;
3620 3E 5D C3 03 60         822         A="]" [INCIX]  RET
3625                        823 ;
3625 00 00                  824 添字DATA  DW $0000
3627                        825 ;
3627                        826 ;
3627                        827 [型倍] ;in A:type,HL:data
3627 11 02 00               828         DE=2
362A FE 02 20 04 1E 01      829         IF A=CHR$  : E=1
3630 18 08 FE 04 20 04 3A   830         EF A=REAL$ : E=(精度)
3637 AE 59 5F
363A                        831 ;       EF A=INT$  : E=2
363A                        832         FI
363A C3 23 6A               833         &[乗算] RET
363D                        834 ;
363D                        835 ;
363D                        836 [配列初期値]
363D FD E5 E1 CD FC 59      837         HL=IY [表DATA変更]
3643                        838 ;
3643 CD C6 51 F5            839         [初期値前ｶｯｺ] PUSH AF
3647                        840 ;
3647 3A 51 5A F5            841         A=(表域) PUSH AF
364B                        842 ;
364B CD F0 56               843         [CODE処理]  :in (登録型)
364E                        844 ;
364E F1 32 51 5A            845         POP AF  (表域)=A
3652                        846 ;       BC:ﾊﾞｲﾄ数
3652 C5                     847         PUSH BC
3653 2A 25 36 3A 52 5A CD   848         HL=(添字DATA) A=(登録型) [型倍]
365A 27 36
365C C1                     849         POP  BC
365D 7C B5 28 15            850         IF HL<>0 THEN
3661 B7 ED 42               851           SUB HL,BC
3664 30 05 CD 14 62         852           IF <  : [DATA過多]
3669 18 0B 28 09 AF CD 7B   853           EF <> : DO HL { A=0 @[IY]A }
3670 5B 2B 7C B5 20 F7
3676                        854           FI
3676                        855         FI
3676                        856 ;
3676 CD E6 51               857         [初期値後ｶｯｺ]
3679 F1 CD B6 51            858         POP AF [後括弧ｴﾗｰ?]
367D C9                     859         RET
367E                        860 ;
367E                        861 ;
367E                        862 [GETWORKﾎﾟｲﾝﾀ]
367E 2A CA 33               863         HL=(WORKﾎﾟｲﾝﾀ)
3681                        864 ;
3681 3A C7 33 FE 00 20 03   865         IF (WORK指定)=FALSE THEN  HL=IY
3688 FD E5 E1
368B                        866 ;
368B C9                     867         RET
368C                        868 ;
368C                        869 ;
368C                        870 [ADDWORKﾎﾟｲﾝﾀ] ;in HL:ﾊﾞｲﾄ
368C 3A C7 33 FE 01 20 0A   871         IF (WORK指定)=TRUE THEN
3693 ED 5B CA 33            872           DE=(WORKﾎﾟｲﾝﾀ)
3697 19 22 CA 33            873           ADD HL,DE  (WORKﾎﾟｲﾝﾀ)=HL
369B 18 09                  874         ELSE
369D AF CD 7B 5B 2B 7C B5   875           DO HL { A=0  @[IY]A }
36A4 20 F7
36A6                        876         FI
36A6                        877 ;
36A6 C9                     878         RET
36A7                        879 ;
36A7                        880 ;
36A7                        881 [名前登録]  :in A:type,HL:data,DE:補助
36A7 F5 D5 E5               882         PUSH AF/DE/HL
36AA                        883 ;
36AA CD 25 3D               884         [名前ﾁｪｯｸ]
36AD                        885 ;
36AD CD 5A 59 DC C8 61      886         [表ｻｰﾁ] IF C [二重宣言!]
36B3                        887 ;
36B3 E1 D1 F1               888         POP  HL/DE/AF
36B6                        889 ;
36B6 C3 94 58               890         [表登録] RET
36B9                        891 ;
36B9                        892 ;
36B9                        893 [ﾌﾟﾛﾄﾀｲﾌﾟ宣言]
36B9 3E 00 32 51 5A         894         (表域)=大域$
36BE 3E 03 32 52 5A         895         (登録型)=INT$
36C3                        896         {
36C3 CD A7 37 30 03 32 52   897           [関数型ｻｰﾁ] IF C THEN (登録型)=A
36CA 5A
36CB CD AC 5B               898           [SPCUT]
36CE                        899 ;
36CE CD 1B 38               900           [関数名登録]
36D1                        901 ;
36D1 2A 18 38 7D D6 08 6F   902           HL=(関数WKﾎﾟｲﾝﾀ) SUB HL,引数最大数
36D8 7C DE 00 67
36DC 22 2A 37               903           (ﾌﾟﾛﾄWKﾎﾟｲﾝﾀ)=HL
36DF AF 32 29 37            904           (ﾌﾟﾛﾄ引数の数)=0
36E3                        905 ;
36E3 CD 3B 58               906           [ｶｯｺ開き]
36E6                        907           {
36E6 CD AC 5B FE 29 28 22   908             [SPCUT] IF A=")" EXIT
36ED                        909 ;
36ED CD 86 37 4F D4 C2 61   910             [型ｻｰﾁ] C=A  IF NC [文法ｴﾗｰ!]
36F4                        911 ;
36F4 3A 29 37 3C 32 29 37   912             INC (ﾌﾟﾛﾄ引数の数)
36FB                        913 ;           A=(ﾌﾟﾛﾄ引数の数)
36FB FE 09 D4 CF 61         914             IF A>引数最大数 [引数過多!]
3700                        915 ;
3700 2A 2A 37               916             HL=(ﾌﾟﾛﾄWKﾎﾟｲﾝﾀ)
3703 79 CD 5B 45            917             A=C  #POKE+
3707 22 2A 37               918             (ﾌﾟﾛﾄWKﾎﾟｲﾝﾀ)=HL
370A                        919 ;
370A CD B7 37               920             [ｶﾝﾏ]
370D 38 D7                  921           } UNTIL NC
370F                        922 ;
370F CD 36 58               923           [ｶｯｺ閉じ]
3712                        924 ;
3712 2A 18 38 7D D6 09 6F   925           HL=(関数WKﾎﾟｲﾝﾀ) SUB HL,引数最大数+1
3719 7C DE 00 67
371D 3A 29 37 CD 9A 1F      926           A=(ﾌﾟﾛﾄ引数の数) #POKE
3723                        927 ;
3723 CD B7 37               928           [ｶﾝﾏ]
3726 38 9B                  929         } UNTIL NC
3728 C9                     930         RET
3729                        931 ;
3729 00                     932 ﾌﾟﾛﾄ引数の数  DB 0
372A 00 00                  933 ﾌﾟﾛﾄWKﾎﾟｲﾝﾀ   DW $0000
372C                        934 ;
372C                        935 ;
372C                        936 [MACHINE宣言]
372C 3E 00 32 51 5A         937         (表域)=大域$
3731 3E 03 32 52 5A         938         (登録型)=INT$
3736                        939         {
3736 CD A7 37 30 03 32 52   940           [関数型ｻｰﾁ] IF C THEN (登録型)=A
373D 5A
373E CD AC 5B               941           [SPCUT]
3741 3E 0C 21 00 00 11 00   942           A=未M関$ HL=0 DE=0 [名前登録]
3748 00 CD A7 36
374C                        943 ;
374C CD 3B 58               944           [ｶｯｺ開き]
374F CD AC 5B               945           [SPCUT]
3752 FE 29 28 18            946           IF A<>")" THEN
3756 CD 17 34               947             [INT定数式!]
3759 7D D6 06 7C DE 00 38   948             IF HL>M引数最大数 THEN
3760 06
3761 CD 1F 62 21 05 00      949               [引数過多]  HL=M引数最大数
3767                        950             FI
3767                        951 ;
3767 EB                     952             EX DE,HL
3768 2A 57 5A               953             HL=(表ﾎﾟｲﾝﾀ)
376B                        954 ;           DEC HL  A=D #POKE   ;補助DATA
376B                        955 ;           DEC HL  A=E #POKE   ;補助DATA
376B CD 03 5A               956             [DEC_POKE_DE]       ;補助DATA
376E                        957           FI
376E CD 36 58               958           [ｶｯｺ閉じ]
3771                        959 ::ｱﾄﾞﾚｽ
3771 CD 23 60 3A 0D         960           [SPSCH] DM ":",$0D
3776 30 08                  961           IF C THEN
3778 CD 22 3A 3E 08 CD 0D   962             [INT定数式] A=M関$ [表変更]
377F 5A
3780                        963           FI
3780                        964 ;
3780 CD B7 37               965           [ｶﾝﾏ]
3783 38 B1                  966         } UNTIL NC
3785 C9                     967         RET
3786                        968 ;
3786                        969 ;
3786                        970 [型ｻｰﾁ]  ;out A:INT$/REAL$
3786 CD 67 60               971         [TBLｻｰﾁ]
3789 49 4E D4 03 00         972         DM "INT"+$80   DW INT$
378E 52 45 41 CC 04 00      973         DM "REAL"+$80  DW REAL$
3794 00                     974         DM 0
3795 7D                     975         A=L
3796 C9                     976         RET
3797                        977 ;
3797                        978 ;
```

▶現在，工場実習にて大阪に長期出張中。腰痛と手先の痛みと戦っています。

戸崎　宗(24)愛知県

```
3797                     979 [型ｻｰﾁ2] :out A:CHR$/INT$/REAL$
3797 CD 86 37 D8         980         [型ｻｰﾁ]  IF C RET
379B                     981 :
379B CD 67 60            982         [TBLｻｰﾁ]
379E 43 48 41 D2 02 00   983         DM "CHAR"+$80  DW CHR$
37A4 00                  984         DM 0
37A5 7D                  985         A=L
37A6 C9                  986         RET
37A7                     987 :
37A7                     988 :
37A7                     989 [関数型ｻｰﾁ] :out A:VOID$/INT$/REAL$
37A7 CD 86 37 D8         990         [型ｻｰﾁ]  IF C RET
37AB                     991 :
37AB CD 67 60            992         [TBLｻｰﾁ]
37AE 56 4F 49 C4 01 00   993         DM "VOID"+$80  DW VOID$
37B4 00                  994         DM 0
37B5 7D                  995         A=L
37B6 C9                  996         RET
37B7                     997 :
37B7                     998 :
37B7                     999 [ｶﾝﾏ]
37B7 CD 23 60 2C 0D     1000         [SPSCH]  DM ",",$0D
37BC C9                 1001         RET
37BD                    1002 :
37BD                    1003 ;************************************
37BD                    1004 :
37BD                    1005 [関数頭書き]
37BD 3E 03 32 52 5A     1006         (登録型)=INT$
37C2 CD A7 37 30 03 32 52 1007       [関数型ｻｰﾁ]  IF C THEN (登録型)=A
37C9 5A
37CA                    1008 :
37CA 3A 52 5A 32 1A 38  1009         (関数型)=(登録型)
37D0                    1010 :
37D0 CD 25 3D           1011         [名前ﾁｪｯｸ]
37D3 CD 82 59           1012         [大域表ｻｰﾁ]
37D6 FE 0C CA 5D 39     1013         IF A=未M関$ [M関数頭書き] RET
37DB                    1014 :
37DB                    1015 :       B:型
37DB B7 20 05           1016         IF A=0 THEN
37DE                    1017 :         (表域)=大域$
37DE CD 18 38           1018           [関数名登録]
37E1                    1019 :
37E1 18 20 FE 0A 20 19  1020         EF A=未関数$ THEN
37E7 3A 52 5A B8 20 11  1021           IF (登録型)<>B THEN
37ED CD 04 62           1022             [型ｴﾗｰ]
37F0                    1023 :
37F0 2A 57 5A           1024             HL=(表ﾎﾟｲﾝﾀ)
37F3 2B 2B              1025             DEC HL/HL              :補助DATA
37F5 2B 2B              1026             DEC HL/HL              :表DATA
37F7 2B 3A 52 5A CD 9A 1F 1027          DEC HL  A=(登録型) #POKE :型
37FE                    1028           FI
37FE                    1029 :
37FE 18 03              1030         ELSE
3800 CD C8 61           1031           [二重宣言!]
3803                    1032         FI
3803                    1033 :
3803 2A 57 5A 22 12 38  1034         (関数表ﾎﾟｲﾝﾀ)=HL=(表ﾎﾟｲﾝﾀ)
3809                    1035 :
3809 CD 38 58 CD 5A 38 C3 1036       [ｶｯｺ開き]  [仮引数]  [ｶｯｺ閉じ] RET
3810 36 58
3812                    1037 :
3812 00 00              1038 関数表ﾎﾟｲﾝﾀ DW $0000
3814 00 00              1039 関数WKTOP   DW $0000
3816 00 00              1040 関数WKEND   DW $0000
3818 00 00              1041 関数WKﾎﾟｲﾝﾀ DW $0000
381A 00                 1042 関数型      DB 0
381B                    1043 :
381B                    1044 :
381B                    1045 [関数名登録] :in (登録型),(表域)
381B CD AC 5B           1046         [SPCUT]
381E ED 53 18 38        1047         DE=(関数WKﾎﾟｲﾝﾀ)
3822 3E 0A 21 00 00 CD A7 1048       A=未関数$  HL=0  [名前登録]
3829 36
382A                    1049 :
382A 2A 18 38           1050         HL=(関数WKﾎﾟｲﾝﾀ)
382D ED 5B 16 38        1051         DE=(関数WKEND)
3831 7B D6 0B 5F 7A DE 00 1052       SUB DE,引数最大数+3
3838 57
3839 7D 93 7C 9A D4 5E 61 1053       IF HL>=DE [関数WKOVER!]
3840                    1054 :       HL=(関数WKﾎﾟｲﾝﾀ)
3840 AF CD 5B 45        1055         A=0    #POKE+       :ﾁｪｲﾝ
3844 AF CD 5B 45        1056         A=0    #POKE+       :ﾁｪｲﾝ
3848 3E FF CD 5B 45     1057         A=NULL #POKE+       :数
384D 06 08              1058         DO B,引数最大数 {
384F 3E FF CD 5B 45     1059           A=NULL  #POKE+  :型
3854 10 F9              1060         }
3856 22 18 38           1061         (関数WKﾎﾟｲﾝﾀ)=HL
3859 C9                 1062         RET
385A                    1063 :
385A                    1064 :
385A                    1065 [仮引数]
385A 21 43 39 22 53 39  1066         (引数ﾒﾓPTR)=HL=引数ﾒﾓ
3860                    1067 :       HL=引数ﾒﾓ
3860 06 10 36 00 23 10 F8 1068       DO B,引数ﾒﾓSIZE { (HL)=0  INC HL }
3867                    1069 :
3867 2A 55 5A 22 57 39  1070         (仮引数ﾎﾟｲﾝﾀ)=HL=(補助DATA)
386D FD 22 59 39        1071         (仮引数のｱﾄﾞﾚｽ)=IY
3871 AF 32 5B 39        1072         (仮引数の数)=0
3875                    1073 :
3875 CD AC 5B           1074         [SPCUT]
3878 FE 29 28 62        1075         IF A<>")" THEN
387C 3E 01 32 51 5A     1076           (表域)=局所$
3881 3E 03 32 52 5A     1077           (登録型)=INT$
3886                    1078           {
3886                    1079 ;引数ﾒﾓ
3886 FD E5 C1           1080             BC=IY
3889 2A 53 39           1081             HL=(引数ﾒﾓPTR)
388C 70 23 71 23        1082             (HL)=B INC HL  (HL)=C INC HL
3890                    1083 :
3890 22 53 39           1084             (引数ﾒﾓPTR)=HL
3893                    1085 ;登録
3893 FD E5 E1 CD 43 35  1086             HL=IY  [変数登録]
3899                    1087 :
3899 06 02 3A 52 5A FE 04 1088           B=2  IF (登録型)=REAL$ THEN B=(精度)
38A0 20 04 3A AE 69 47
38A6 AF CD 7B 5B 10 FA  1089             DO B { A=0  @[IY]A }
38AC                    1090 ;数ﾁｪｯｸ
38AC 3A 5B 39 3C 32 5B 39 1091           INC (仮引数の数)
38B3                    1092 :           A=(仮引数の数)
38B3 FE 09 D4 CF 61     1093             IF A>引数最大数 [引数過多!]
38B8                    1094 ;型ﾁｪｯｸ
38B8 2A 57 39           1095             HL=(仮引数ﾎﾟｲﾝﾀ)
38BB 23 23              1096             INC HL/HL
38BD 3A 5B 39 5F 16 00  1097             E=(仮引数の数) D=0
38C3 19                 1098             ADD HL,DE
38C4 CD 94 1F 47 3A 52 5A 1099           #PEEK  B=A  A=(登録型) #POKE
38CB CD 9A 1F
38CE                    1100 :           A=(登録型)
38CE B8 28 08 78 FE FF 28 1101           IF A<>B AND B<>NULL THEN
38D5 03
38D6 CD 35 62           1102               [引数型ｴﾗｰ]
38D9                    1103             FI
38D9                    1104 :
38D9 CD B7 37           1105             [ｶﾝﾏ]
38DC 38 A8              1106           } UNTIL NC
38DE                    1107         FI
38DE 2A 53 39 22 55 39  1108         (引数ﾒﾓEND)=HL=(引数ﾒﾓPTR)
38E4                    1109 :
38E4                    1110 ;引数ﾁｪｲﾝ
38E4                    1111 :
38E4 3A 5B 39 B7 28 3E  1112         IF (仮引数の数)<>0 THEN
38EA 2A 57 39           1113           HL=(仮引数ﾎﾟｲﾝﾀ)
38ED ED 4B 59 39        1114           BC=(仮引数のｱﾄﾞﾚｽ)
38F1 CD 94 1F 5F 79 CD 5B 1115         #PEEK  E=A  A=C #POKE+
38F8 45
38F9 CD 94 1F 57 78 CD 9A 1116         #PEEK  D=A  A=B #POKE
3900 1F
3901 EB                 1117           EX DE,HL
3902 7C B5 28 22        1118           UNTIL HL=0 {
3906 ED 4B 53 39        1119             BC=(引数ﾒﾓPTR)
390A 78 FE 39 20 03 79 FE 1120           IF BC=引数ﾒﾓ THEN BC=(引数ﾒﾓEND)
3911 43 20 04 ED 4B 55 39
3918                    1121 :
3918 CD 78 69           1122             CALL @PATCH2
391B                    1123 :           DE=(OFFSET) ADD HL,DE
391B                    1124 :           DEC BC  E=(HL) A=(BC)
391B 77 23              1125             (HL)=A  INC HL
391D 0B 56 0A 77        1126             DEC BC  D=(HL) (HL)=(BC)
3921 ED 43 53 39        1127             (引数ﾒﾓPTR)=BC
3925 EB                 1128             EX DE,HL
3926 18 DA              1129           }
3928                    1130         FI
3928                    1131 :
3928                    1132 ;仮引数の数
3928                    1133 :
3928 2A 57 39           1134         HL=(仮引数ﾎﾟｲﾝﾀ)
392B                    1135 :       HL:関数WK
392B 23 23 CD 94 1F 47  1136         INC HL/HL     #PEEK  B=A
3931 3A 5B 39 CD 9A 1F  1137         A=(仮引数の数) #POKE
3937                    1138 :       A=(仮引数の数)
3937 B8 28 08 78 FE FF 28 1139       IF A<>B AND B<>NULL THEN [引数数ｴﾗｰ]
393E 03 CD 27 62
3942                    1140 :
3942 C9                 1141         RET
3943                    1142 :
```

```
3943                    1143 :
0010                    1144 引数ﾒﾓSIZE    EQU 引数最大数*2
3943 00 00 00 00 00 00 00 1145 引数ﾒﾓ     DS  引数ﾒﾓSIZE
394A 00 00 00 00 00 00 00
3951 00 00
3953 00 00              1146 引数ﾒﾓPTR     DW  $0000
3955 00 00              1147 引数ﾒﾓEND     DW  $0000
3957                    1148 :
3957 00 00              1149 仮引数ﾎﾟｲﾝﾀ   DW $0000
3959 00 00              1150 仮引数のｱﾄﾞﾚｽ DW $0000
395B 00 00              1151 仮引数の数     DW $0000
395D                    1152 :
395D                    1153 :
395D                    1154 [M関数頭書き]
395D                    1155 :       [名前ﾁｪｯｸ] [大域表ｻｰﾁ]
395D F5                 1156         PUSH AF
395E                    1157 :
395E 2A 57 5A 22 12 38  1158         (関数表ﾎﾟｲﾝﾀ)=HL=(表ﾎﾟｲﾝﾀ)
3964                    1159 :
3964 CD 38 58           1160         [ｶｯｺ開き]
3967                    1161 :
3967 CD AC 5B 21 00 00 FE 1162       [SPCUT]  HL=0  IF A<>")" [INT定数式!]
396E 29 C4 17 34
3972 7D D6 06 7C DE 00 38 1163       IF HL>M引数最大数 THEN
3979 06
397A CD 1F 62 21 05 00  1164           [引数過多]  HL=M引数最大数
3980                    1165         FI
3980 EB                 1166         EX DE,HL
3981                    1167 :       DE:引数の数
3981 F1                 1168         POP AF
3982 FE 0C 20 18        1169         IF A=未M関$ THEN
3986 2A 57 5A           1170           HL=(表ﾎﾟｲﾝﾀ)
3989 2B CD 94 1F 47     1171           DEC HL  #PEEK B=A   :補助DATA
398E 2B CD 94 1F 4F     1172           DEC HL  #PEEK C=A   :補助DATA
3993                    1173 :
3993 D5                 1174           PUSH DE
3994 7A B8 20 02 78 B9 C4 1175         IF DE<>BC [引数数ｴﾗｰ]
399B 27 62
399D D1                 1176           POP  DE
399E                    1177         FI
399E                    1178 :       DE:引数の数
399E 2A 57 5A           1179         HL=(表ﾎﾟｲﾝﾀ)
39A1                    1180 :       DEC HL  A=D #POKE     :補助DATA
39A1                    1181 :       DEC HL  A=E #POKE     :補助DATA
39A1 CD 03 5A           1182         [DEC_POKE_DE]         :補助DATA
39A4                    1183 :
39A4 C3 36 58           1184         [ｶｯｺ閉じ] RET
39A7                    1185 :
39A7                    1186 :
39A7                    1187 [関数LINK]
39A7 2A 12 38 FD E5 C1  1188         HL=(関数表ﾎﾟｲﾝﾀ)  BC=IY
39AD 2B                 1189         DEC HL                        :補助DATA
39AE 2B                 1190         DEC HL                        :補助DATA
39AF 2B CD 94 1F 57 78 CD 1191       DEC HL #PEEK D=A  A=B #POKE :表DATA
39B6 9A 1F
39B8 2B CD 94 1F 5F 79 CD 1192       DEC HL #PEEK E=A  A=C #POKE :表DATA
39BF 9A 1F
39C1 2B                 1193         DEC HL                       :型
39C2 2B CD 94 1F 3D CD 9A 1194       DEC HL #PEEK  DEC A   #POKE :TYPE
39C9 1F
39CA                    1195 :
39CA EB                 1196         EX DE,HL
39CB FD E5 C1           1197         BC=IY
39CE 7C B5 28 0F        1198         UNTIL HL=0 {
39D2                    1199 :
39D2 CD 81 69           1200           CALL @PATCH1
39D5                    1201 :         DE=(OFFSET) ADD HL,DE
39D5                    1202 :         E=(HL) (HL)=C  INC HL
39D5 56 70              1203           D=(HL) (HL)=B
39D7                    1204 :         PUSH DE
39D7 3A E2 39 3D 32 E2 39 1205         DEC (未ｺｰﾙ回数)
39DE                    1206 :         POP  DE
39DE EB                 1207           EX DE,HL
39DF 18 ED              1208         }
39E1 C9                 1209         RET
39E2                    1210 :
39E2 00                 1211 未ｺｰﾙ回数 DB 0
39E3                    1212 :
39E3                    1213 ;************************************
39E3                    1214 :
39E3                    1215         NEXTFILE REAL-2
39E3                    1215+        $CHAIN   REAL-2    03A.Asm
39E3                       1 :
39E3                       2 :         実数型コンパイラ  REAL
39E3                       3 :
39E3                       4 :           REAL-2    03A.Asm
39E3                       5 :
39E3                       6 :         [式]
39E3                       7 :
39E3                       8 ;************************************
39E3                       9 :
39E3                      10 [式初期化]
39E3 CD 41 3C             11         [変換SW初期化]
39E6 CD 70 50             12         [式ｽﾀｯｸ初期化]    :for 演算
39E9 CD 6A 43             13         [定数ｽﾀｯｸ初期化]  :for 実数定数演算
39EC C3 BB 4F             14         [TEMP初期化]  RET :for 実数演算TEMP
39EF                      15 :
39EF                      16 [CODE_INT]
39EF CD 22 3A C3 76 5B    17         [INT定数式] @[IY]HL RET
39F5                      18 :
39F5                      19 [CODE_REAL]
39F5                      20 :
39F5                      21 ;REAL定数式
39F5 3A 51 5A F5          22         A=(表域)     PUSH AF
39F9 2A 57 5A E5          23         HL=(表ﾎﾟｲﾝﾀ) PUSH HL
39FD                      24 :
39FD CD 70 3A CD 64 4D    25         [論理項] [TOP定数実数化]
3A03                      26 :
3A03 E1 22 57 5A          27         POP HL  (表ﾎﾟｲﾝﾀ)=HL
3A07 F1 32 51 5A          28         POP AF  (表域)=A
3A0B                      29 :
3A0B CD B4 50 FE 01 C4 DD 30         [POP_1]  IF A<>定数% [定数ｴﾗｰ]
3A12 61
3A13 22 5F 44             31         (定数ﾎﾟｲﾝﾀ)=HL
3A16                      32 :
3A16                      33 :
3A16 3A AE 69 47          34         DO B,(精度) {
3A1A 7E 23 CD 7B 5B       35           A=(HL) INC HL  @[IY]A
3A1F 10 F9                36         }
3A21 C9                   37         RET
3A22                      38 :
3A22                      39 :
3A22                      40 [INT定数式] ;out A:type%,HL:data,(型)
3A22 3A 51 5A F5          41         A=(表域)     PUSH AF
3A26 2A 57 5A E5          42         HL=(表ﾎﾟｲﾝﾀ) PUSH HL
3A2A                      43 :
3A2A CD 70 3A             44         [論理項]
3A2D                      45 :
3A2D E1 22 57 5A          46         POP HL  (表ﾎﾟｲﾝﾀ)=HL
3A31 F1 32 51 5A          47         POP AF  (表域)=A
3A35                      48 :
3A35 CD B4 50             49         [POP_1]
3A38 F5                   50         PUSH AF
3A39 FE 01 C4 DD 61       51         IF A<>定数%  [定数ｴﾗｰ]
3A3E 3A 6F 50 FE 04 CC 0B 52         IF (型)=REAL$ [REAL不可]
3A45 62
3A46 F1                   53         POP  AF
3A47 C9                   54         RET
3A48                      55 :
3A48                      56 :
3A48                      57 [初式]
3A48 CD E3 39             58         [式初期化]
3A4B                      59 :       [式]
3A4B                      60 :       RET
3A4B                      61 :
3A4B                      62 [式]
3A4B CD 54 3A C3 D6 4B    63         [論理式] [ｺｰﾄﾞ生成1] RET
3A51                      64 :
3A51                      65 :
3A51                      66 [初論理式]
3A51 CD 7C 50             67         [初ｺｰﾄﾞ初期化]
3A54                      68 :       [論理式]
3A54                      69 :       RET
3A54                      70 :
3A54                      71 [論理式]
3A54 CD 70 3A CD B7 37 D0 72         [論理項] [ｶﾝﾏ] IF NC RET
3A5B                      73 :
3A5B CD D6 4B             74         [ｺｰﾄﾞ生成1]
3A5E                      75         {
3A5E CD B4 50             76           [POP_1]
3A61                      77 :
3A61 CD 6D 3A CD D6 4B    78           [初論理項] [ｺｰﾄﾞ生成1]
3A67                      79 :
3A67 CD B7 37             80           [ｶﾝﾏ]
3A6A 38 F2                81         } UNTIL NC
3A6C                      82 :
3A6C C9                   83         RET
3A6D                      84 :
3A6D                      85 :
3A6D                      86 [初論理項] ;for 引数/定数式
3A6D CD 7C 50             87         [初ｺｰﾄﾞ初期化]
3A70                      88 :       [論理項]
3A70                      89 :       RET
3A70                      90 :
3A70                      91 [論理項]
```

```
3A70 CD 90 3A             92         [論理AND項]
3A73                      93         {
3A73 CD 67 60             94           [TBLｻｰﾁ]
3A76 7C 7C 0D 00 00       95           DM "||",$0D  DW 0
3A7B 21 21 0D 00 00       96           DM "!!",$0D  DW 0
3A80 00                   97           DM 0
3A81 D0                   98           IF NC RET
3A82                      99 :
3A82 CD 90 3A            100           [論理AND項]
3A85                     101 :
3A85 21 3B 01            102           HL=&実[論理OR]-&R
3A88 11 3F 01 CD 0E 4B   103           DE=&整[論理OR]-&R  [二項演算2INT]
3A8E 18 E3               104         }
3A90                     105 :
3A90                     106 :
3A90                     107 [論理AND項]
3A90 CD A8 3A            108         [ﾋﾞｯﾄ式]
3A93                     109         {
3A93 CD 23 60 26 26 0D D0 110          [SPSCH] DM "&&",$0D  IF NC RET
3A9A                     111 :
3A9A CD A8 3A            112           [ﾋﾞｯﾄ式]
3A9D                     113 :
3A9D 21 45 01            114           HL=&実[論理AND]-&R
3AA0 11 4F 01 CD 0E 4B   115           DE=&整[論理AND]-&R  [二項演算2INT]
3AA6 18 EB               116         }
3AA8                     117 :
3AA8                     118 :
3AA8                     119 [ﾋﾞｯﾄ式]
3AA8 CD CB 3A            120         [関係式]
3AAB                     121         {
3AAB CD 67 60            122           [TBLｻｰﾁ]
3AAE 4F D2 80 02         123           DM "OR"+$80   DW &[ﾋﾞｯﾄOR]-&R
3AB2 41 4E C4 9C 02      124           DM "AND"+$80  DW &[ﾋﾞｯﾄAND]-&R
3AB7 58 4F D2 25 01      125           DM "XOR"+$80  DW &[ﾋﾞｯﾄXOR]-&R
3ABC 00                  126           DM 0
3ABD D0                  127           IF NC RET
3ABE                     128 :
3ABE E5                  129           PUSH HL
3ABF CD CB 3A            130             [関係式]
3AC2 D1                  131           POP  DE
3AC3 21 00 00            132           HL=0          ;REAL型は不可
3AC6 CD 27 4B            133           [二項演算3]
3AC9 18 E0               134         }
3ACB                     135 :
3ACB                     136 :
3ACB                     137 [関係式]
3ACB CD 21 3B            138         [ｼﾌﾄ式]
3ACE                     139         {
3ACE CD 67 60            140           [TBLｻｰﾁ]
3AD1 3D 3D 0D 09 3B      141           DM "==",$0D  DW TBL[==]
3AD6 3C 3E 0D 0D 3B      142           DM "<>",$0D  DW TBL[<>]
3ADB 21 3D 0D 0D 3B      143           DM "!=",$0D  DW TBL[<>]
3AE0 3C 3D 0D 11 3B      144           DM "<=",$0D  DW TBL[<=]
3AE5 3E 3D 0D 15 3B      145           DM ">=",$0D  DW TBL[>=]
3AEA 3C 0D 19 3B         146           DM "<",$0D   DW TBL[<]
3AEE 3E 0D 1D 3B         147           DM ">",$0D   DW TBL[>]
3AF2 00                  148           DM 0
3AF3 D0                  149           IF NC RET
3AF4                     150 :
3AF4 E5                  151           PUSH HL
3AF5 CD 21 3B            152             [ｼﾌﾄ式]
3AF8 C1                  153           POP  BC
3AF9 0A 5F 03            154           L=(BC) INC BC
3AFC 0A 57 03            155           H=(BC) INC BC
3AFF 0A 5F 03            156           E=(BC) INC BC
3B02 0A 57               157           D=(BC)
3B04 CD 0E 4B            158           [二項演算2INT]
3B07 18 C5               159         }
3B09 64 01 69 01         160 TBL[==]  DW &実[==]-&R,&整[==]-&R
3B0D 56 01 5B 01         161 TBL[<>]  DW &実[<>]-&R,&整[<>]-&R
3B11 7A 01 80 01         162 TBL[<=]  DW &実[<=]-&R,&整[<=]-&R
3B15 7B 01 81 01         163 TBL[>=]  DW &実[>=]-&R,&整[>=]-&R
3B19 8B 01 91 01         164 TBL[<]   DW &実[<]-&R,  &整[<]-&R
3B1D 8A 01 90 01         165 TBL[>]   DW &実[>]-&R,  &整[>]-&R
3B21                     166 :
3B21                     167 :
3B21                     168 [ｼﾌﾄ式]
3B21 CD 40 3B            169         [加減項]
3B24                     170         {
3B24 CD 67 60            171           [TBLｻｰﾁ]
3B27 3C 3C 0D 35 01      172           DM "<<",$0D  DW &[<<]-&R
3B2C 3E 3E 0D 2C 01      173           DM ">>",$0D  DW &[>>]-&R
3B31 00                  174           DM 0
3B32 D0                  175           IF NC RET
3B33                     176 :
3B33 E5                  177           PUSH HL
3B34 CD 40 3B            178             [加減項]
3B37 D1                  179           POP  DE
3B38 21 00 00            180           HL=0          ;REAL型は不可
3B3B CD 2C 4B            181           [二項演算2]
3B3E 18 E4               182         }
3B40                     183 :
3B40                     184 :
3B40                     185 [加減項]
3B40 CD 5A 3B            186         [乗除項]
3B43                     187         {
3B43 CD 67 60            188           [TBLｻｰﾁ]
3B46 2B 0D F8 48         189           DM "+",$0D  DW [加算処理]
3B4A 2D 0D 04 49         190           DM "-",$0D  DW [減算処理]
3B4E 00                  191           DM 0
3B4F D0                  192           IF NC RET
3B50                     193 :
3B50 E5 CD 5A 3B E1      194           PUSH HL  [乗除項]  POP HL
3B55                     195 :
3B55 CD 81 1F            196           CALL [HL]
3B58 18 E9               197         }
3B5A                     198 :
3B5A                     199 :
3B5A                     200 [乗除項]
3B5A CD 81 3B            201         [項]
3B5D                     202         {
3B5D CD 67 60            203           [TBLｻｰﾁ]
3B60 2A 0D 10 49         204           DM "*",$0D  DW [乗算処理]
3B64 2F 0D 1C 49         205           DM "/",$0D  DW [除算処理]
3B68 25 0D 78 3B         206           DM "%",$0D  DW [剰余処理]
3B6C 00                  207           DM 0
3B6D D0                  208           IF NC RET
3B6E                     209 :
3B6E E5 CD 81 3B E1      210           PUSH HL  [項]  POP HL
3B73                     211 :
3B73 CD 81 1F            212           CALL [HL]
3B76 18 E5               213         }
3B78                     214 :
3B78                     215 [剰余処理]
3B78 21 2F FF            216         HL=#MOD+$FF00
3B7B 11 09 01 C3 2C 4B   217         DE=&[剰余]-&R  [二項演算2] RET
3B81                     218 :
3B81                     219 :
3B81                     220 [項]
3B81 CD 67 60            221         [TBLｻｰﾁ]
3B84 43 56 49 54 D2 08 4D 222        DM "CVITR"+$80  DW [ｷｬｽﾄITR]
3B8B 43 56 55 54 D2 2F 3C 223        DM "CVUTR"+$80  DW [ｷｬｽﾄUTR]
3B92 43 56 52 54 C9 2E 4D 224        DM "CVRTI"+$80  DW [ｷｬｽﾄRTI]
3B99 43 56 52 54 D5 39 3C 225        DM "CVRTU"+$80  DW [ｷｬｽﾄRTU]
3BA0 4E 4F D4 48 3C      226         DM "NOT"+$80    DW [NOT演算子]
3BA5 43 50 CC 51 3C      227         DM "CPL"+$80    DW [CPL演算子]
3BAA 41 42 D3 5A 3C      228         DM "ABS"+$80    DW [ABS演算子]
3BAF 53 47 CE 63 3C      229         DM "SGN"+$80    DW [SGN演算子]
3BB4 00                  230         DM 0
3BB5 30 04               231         IF C THEN
3BB7 E5 C3 6C 3C         232           PUSH HL  [因子] RET
3BBB                     233 :         [因子]
3BBB                     234 :         POP  HL
3BBB                     235 :         CALL [HL]
3BBB                     236 :         RET
3BBB                     237         FI
3BBB                     238 :
3BBB CD AC 5B            239         [SPCUT]
3BBE FE 21 20 0F DD 7E 01 240        IF A="!" AND (IX+1)<>"!" THEN
3BC5 FE 21 28 08
3BC9 DD 23               241           INC IX
3BCB CD 6C 3C C3 48 3C   242           [因子] [NOT演算子] RET
3BD1                     243         FI
3BD1                     244 :
3BD1 CD AC 5B            245         [SPCUT]
3BD4 FE 2D 20 44 DD 7E 01 246        IF A="-" AND (IX+1)<>"-" THEN
3BDB FE 2D 28 3D
3BDF DD 23               247           INC IX
3BE1 CD 6C 3C            248           [因子]
3BE4                     249 ;for -32769 ~ -65535
3BE4 CD A2 50            250           [TOP_ﾏｰｸ]
3BE7 FE 01 20 28 3A 6F 50 251          IF A=定数% AND (型)=INT$ THEN
3BEE FE 03 20 21
3BF2 7D D6 01 7C DE 80 38 252            IF HL>32768 THEN
3BF9 19
3BFA CD B4 50            253               [POP_1]
3BFD E5                  254               PUSH HL
3BFE 3A AE 69 CD 36 44   255               A=(精度) [INC定数ﾎﾟｲﾝﾀ]
3C04                     256 :             HL=(定数ﾎﾟｲﾝﾀ)
3C04 D1 01 14 00 CD 87 4F 257              POP DE  BC=#CVITF CALL[SRB]
3C0B                     258 :             HL=(定数ﾎﾟｲﾝﾀ)
3C0B 3E 04 32 6F 50 C3 CE 259              (型)=REAL$ [PUSH_定数%HL]
3C12 4B
3C13                     260             FI
3C13                     261           FI
```

▶レーザーディスクプレイヤーもないのに「炎の転校生」を買ってしまった。
広瀬　良一(19)茨城県

```
3C13 21 35 FF           262         HL=#NEG+$FF00
3C16 11 1D 01 C3 CE 4A  263         DE=&[単項－]-&R [単項演算1] RET
3C1C                    264         FI
3C1C                    265 :
3C1C CD AC 5B           266         [SPCUT]
3C1F FE 2B 20 09 DD 7E 01 267       IF A="-" AND (IX+1)<>"-" THEN
3C26 FE 2B 28 02
3C2A DD 23              268           INC IX
3C2C                    269         FI
3C2C                    270 :
3C2C C3 6C 3C           271         [因子] RET
3C2F                    272 :
3C2F                    273 [ｷｬｽﾄUTR]
3C2F 3E 55 32 47 3C C3 08 274       (変換SW)="U" [TOP実数化]
3C36 4D
3C37 18 08              275         JR [変換SW初期化]
3C39                    276 :
3C39                    277 [ｷｬｽﾄRTU]
3C39 3E 55 32 47 3C C3 2E 278       (変換SW)="U" [TOP整数化]
3C40 4D
3C41                    279         JR [変換SW初期化]
3C41                    280 :
3C41                    281 [変換SW初期化]
3C41 3E 49 32 47 3C C9  282         (変換SW)="I" RET
3C47                    283 :
3C47 49                 284 変換SW DB "I"
3C48                    285 :
3C48                    286 [NOT演算子]
3C48 21 76 01           287         HL=&実[NOT]-&R
3C4B 11 72 01 C3 F1 4A  288         DE=&整[NOT]-&R [単項演算1INT] RET
3C51                    289 :
3C51                    290 [CPL演算子]
3C51 21 00 00           291         HL=0 ;REALは不可
3C54 11 1E 01 C3 CE 4A  292         DE=&[CPL]-&R [単項演算1]  RET
3C5A                    293 :
3C5A                    294 [ABS演算子]
3C5A 21 62 FF           295         HL=#ABS+$FF00
3C5D 11 1A 01 C3 CE 4A  296         DE=&[ABS]-&R [単項演算1]  RET
3C63                    297 :
3C63                    298 [SGN演算子]
3C63 21 65 FF           299         HL=#SGN+$FF00
3C66 11 0E 01 C3 CE 4A  300         DE=&[SGN]-&R [単項演算1]  RET
3C6C                    301 :
3C6C                    302 :
3C6C                    303 [因子]
3C6C                    304 :(式)
3C6C CD 23 60 28 0D     305         [SPSCH] DM "(",$0D
3C71 30 05              306         IF C THEN
3C73 CD 54 3A C3 35 58  307           [論理式] [ｶｯｺ閉じ] RET
3C79                    308         FI
3C79                    309 :
3C79 CD 67 60           310         [TBLｻｰﾁ]
3C7C 22 0D A5 45        311         DM """",$0D  DW [文字列]
3C80 26 0D 40 3C        312         DM "&",$0D  DW [&ADDR]
3C84 2B 2B 0D 5E 3D     313         DM "++",$0D DW [++前置演算子]
3C89 2D 2D 0D 5B 3D     314         DM "--",$0D DW [--前置演算子]
3C8E 00                 315         DM 0
3C8F DA 81 1F           316         IF C  CALL [HL] RET
3C92                    317 :
3C92                    318 :定数
3C92 CD 71 43 DA CE 4B  319         [定数]  IF C [PUSH_定数%HL] RET
3C98                    320 :
3C98 CD 25 3D           321         [名前ﾁｪｯｸ]
3C9B                    322 :変数/記号定数/配列名
3C9B FE 01 CA BC 3F     323         IF A=名前0 [変数&記定] RET
3CA0                    324 :配列
3CA0 FE 02 CA 70 40     325         IF A=配列0 [配列] RET
3CA5                    326 :関数
3CA5 C3 90 46           327         [関数ｺｰﾙ] RET
3CA8                    328 :
3CA8                    329 :
3CA8                    330 [&ADDR]
3CA8 CD 25 3D           331         [名前ﾁｪｯｸ]
3CAB FE 02 CA D7 3C     332         IF A=関数0 [&関数]    RET
3CB0 FE 03 CA BB 40     333         IF A=配列0 [配列処理] RET
3CB5                    334 :
3CB5 CD 62 59 D4 82 59  335         [局所表ｻｰﾁ] IF NC [大域表ｻｰﾁ]
3CBB                    336 :
3CBB B7 CA 0E 3D        337         IF A=0      [&ﾗﾍﾞﾙ] RET
3CBF FE 0A CA E0 3D     338         IF A=未ﾗﾍﾞﾙ$ [&ﾗﾍﾞﾙ] RET
3CC4 FE 01 28 04 FE 09 38 339       IF A=記定$ OR A>=関数$ THEN [文法ｴﾗｰ]
3CCB 03 CD D6 61
3CCF                    340 ;変数$.記変1$.関変1$.配列1$.配列2$.ﾗﾍﾞﾙ$
3CCF 3E 03 32 6F 50 C3 CE 341       (型)=INT$  [PUSH_定数%HL] RET
3CD6 4B
3CD7                    342 :
3CD7                    343 :
3CD7                    344 [&関数]
3CD7 CD 82 59           345         [大域表ｻｰﾁ]
3CDA FE 09 28 04 FE 03 20 346       IF A=関数$ OR A=M関$ THEN
3CE1 04
3CE2 3E 01              347           A=定数%
3CE4                    348 :
3CE4 18 18              349         ELSE
3CE6 B7 20 08           350           IF A=0 THEN
3CE9                    351 :           (表域)=大域$
3CE9 3E 03 32 52 5A CD 18 352           (登録型)=INT$  [関数名登録]
3CF0 38
3CF1                    353           FI
3CF1 CD 91 4F 3E 21 CD 7B 354         @PUSH_HL  A=@HL @[IY]A  @[IY]関数
3CF8 5B CD 1D 48
3CFC 3E 00              355           A=中間%
3CFE                    356         FI
3CFE                    357 :
3CFE F5 3E 03 32 6F 50 F1 358       PUSH AF  (型)=INT$  POP AF  [PUSH_A%HL]
3D05 CD BF 50
3D08                    359 :
3D08 CD 39 58 C3 36 58  360         [ｶｯｺ開き] [ｶｯｺ閉じ] RET
3D0E                    361 :
3D0E                    362 :
3D0E                    363 [&ﾗﾍﾞﾙ]
3D0E F5 E5              364         PUSH AF/HL
3D10                    365 :
3D10 CD 91 4F 3E 21 CD 7B 366       @PUSH_HL  A=@HL @[IY]A
3D17 5B
3D18                    367 :
3D18 E1 F1 CD 22 56     368         POP  HL/AF  @[IY]ﾗﾍﾞﾙ
3D1D                    369 :
3D1D 3E 03 32 6F 50 C3 BD 370       (型)=INT$  [PUSH_中間%] RET
3D24 50
3D25                    371 :
3D25                    372 :
3D25                    373 [名前ﾁｪｯｸ]  :out A:type0,C:char
3D25 CD 2C 3D D4 C2 61  374         [名前?]  IF NC [文法ｴﾗｰ]
3D2B C9                 375         RET
3D2C                    376 :
3D2C                    377 :
3D2C                    378 [名前?]     :out A:type0,C:char
3D2C AF 32 42 61        379         (漢ﾌﾗｸﾞ)=0
3D30                    380 :
3D30 CD AC 5B DD E5 D1  381         [SPCUT]  DE=IX
3D36                    382 :       A=(DE)             RCF
3D36 CD F1 60 38 04 AF 0E 383       [英字ﾁｪｯｸ]  IF NC THEN  A=0  C=0  RET
3D3D 00 C9
3D3F                    384         {
3D3F 13 1A CD E6 60     385           INC DE   A=(DE) [英数ﾁｪｯｸ]
3D44 38 F9              386         } UNTIL NC
3D46                    387 :       A=(DE)
3D46 4F                 388         C=A
3D47 FE 28 20 04 3E 02  389         IF A="(" : A=関数0
3D4D 18 0A FE 5B 20 04 3E 390       EF A="[" : A=配列0
3D54 03
3D55 18 02 3E 01        391         ELSE     : A=名前0 :変数/記定/ﾗﾍﾞﾙ
3D59                    392         FI
3D59                    393 :A:type0  C:char
3D59 37                 394         SCF
3D5A C9                 395         RET
3D5B                    396 :
3D5B                    397 :
3D5B 3E 2B 11           398 [－－前置演算子]  A=－$  DB SKIP
3D5E 3E 23              399 [＋＋前置演算子]  A=＋$
3D60 32 BB 3F           400         (＋－FLG)=A
3D63 CD 25 3D           401         [名前ﾁｪｯｸ]
3D66 FE 03 CA 01 3E     402         IF A=配列0 [＋－配列] RET
3D6B                    403 :       IF A=名前0 [＋－変数] RET
3D6B                    404 :
3D6B                    405 [＋－変数]
3D6B CD 62 59           406         [局所表ｻｰﾁ]
3D6E D4 82 59           407         IF NC [大域表ｻｰﾁ]
3D71 D4 16 40           408         IF NC [未宣言変数]
3D74                    409 :
3D74 FE 02 20 03 78     410         IF A=変数$             : A=B
3D79 18 11 FE 03 28 04 FE 411       EF A=関変1$ OR A=関変2$ : A=INT$
3D80 04 20 04 3E 03
3D85 18 05 CD FB 61 3E 03 412       ELSE        : [名前誤用]  A=INT$
3D8C                    413         FI
3D8C 32 6F 50           414         (型)=A
3D8F                    415 :
3D8F 3E 02 CD BF 50     416         A=変数% [PUSH_A%HL]
3D94                    417 :       [＋－変数処理]
3D94                    418 :       RET
3D94                    419 :
3D94                    420 [＋－変数処理]
3D94 CD A2 50           421         [TOP_ﾏｰｸ]
3D97 3A 6F 50 FE 03 20 1D 422       IF (型)=INT$ THEN
3D9E CD 91 4F           423           @PUSH_HL
3DA1 CD D8 50 3E 2A CD 73 424         [TOP_DATA] A=@HL[]  @[IY]AHL
3DA8 5B
3DA9 3A BB 3F CD 7B 5B  425           A=(＋－FLG) @[IY]A   :: INC/DEC HL
3DAF CD B4 50 3E 22 CD 73 426         [POP_1]    A=@[]HL  @[IY]AHL
3DB6 5B
3DB7 3E 03              427           A=INT$
3DB9                    428 :
3DB9 18 25              429         ELSE
3DBB 3E 03 32 6F 50 3E 01 430         (型)=INT$  A=定数%  HL=1 [PUSH_A%HL]
3DC2 21 01 00 CD BF 50
3DC8 CD 45 4C CD B4 50  431           [TEMPなしｺｰﾄﾞ生成2] [POP_1]
3DCE 21 20 00           432           HL=#ADD
3DD1 3A BB 3F FE 2B 20 03 433         IF (＋－FLG)=－$ THEN HL=#SUB
3DD8 21 23 00
3DDB CD 7D 4F           434           @[IY]call_SRB
3DDE 3E 04              435           A=REAL$
3DE0                    436         FI
3DE0 32 6F 50           437         (型)=A
3DE3                    438 :       [PUSH_ｺｰﾄﾞ%]
3DE3                    439 :       RET
3DE3                    440 :
3DE3                    441 [PUSH_ｺｰﾄﾞ%]
3DE3 3A 6F 50 FE 03 20 04 442       IF (型)=INT$ : A=中間%
3DEA 3E 00
3DEC 18 02 3E 03        443         ELSE         : A=ｺｰﾄﾞ%
3DF0                    444         FI
3DF0 C3 BF 50           445         [PUSH_A%HL]  RET
3DF3                    446 :
3DF3                    447 :
3DF3                    448 [変数＋－]  :in A:＋$/－$.
3DF3 32 BB 3F           449         (＋－FLG)=A
3DF6                    450 :
3DF6 CD 94 3D           451         [＋－変数処理]
3DF9                    452 :
3DF9 3E 03 32 48 41 C3 5B 453       (配列型)=INT$ [配列＋－後処理] RET
3E00 3F
3E01                    454 :
3E01                    455 :
3E01                    456 [＋－配列]
3E01 CD 67 60           457         [TBLｻｰﾁ]
3E04 4D 45 CD 3F 3E     458         DM "MEM"+$80   DW [＋－MEM]
3E09 4D 45 4D D7 42 3E  459         DM "MEMW"+$80  DW [＋－MEMW]
3E0F 4D 45 4D D2 45 3E  460         DM "MEMR"+$80  DW [＋－MEMR]
3E15 50 4F 52 D4 A8 3E  461         DM "PORT"+$80  DW [＋－PORT]
3E1B 50 4F 52 54 D7 A5 3E 462       DM "PORTW"+$80 DW [＋－PORTW]
3E22 53 4F D3 F7 3E     463         DM "SOS"+$80   DW [＋－SOS]
3E27 53 4F 53 D7 F4 3E  464         DM "SOSW"+$80  DW [＋－SOSW]
3E2D 00                 465         DM 0
3E2E DA 81 1F           466         IF C CALL [HL]  RET
3E31                    467 :
3E31 CD BB 40 3A 49 41 FE 468       [配列処理] IF (代入FLG)=FALSE [文法ｴﾗｰ]
3E38 00 CC D6 61
3E3C                    469 :
3E3C C3 4D 3E           470         [＋－MEM処理] RET
3E3F                    471 :
3E3F                    472 :
3E3F 3E 02 11           473 [＋－MEM]  A=CHR$  DB SKIP
3E42 3E 03 11           474 [＋－MEMW] A=INT$  DB SKIP
3E45 3E 04              475 [＋－MEMR] A=REAL$
3E47 32 48 41           476         (配列型)=A
3E4A                    477 :
3E4A CD F5 42           478         配列用[式]
3E4D                    479 :       [＋－MEM処理]
3E4D                    480 :       RET
3E4D                    481 :
3E4D                    482 [＋－MEM処理]
3E4D 3A 48 41           483         A=(配列型)
3E50 FE 02 20 1C        484         IF A=CHR$ THEN
3E54 CD D6 4B           485           [ｺｰﾄﾞ生成1]
3E57 3A BB 3F FE 23 20 04 486         IF (＋－FLG)=＋$ : DB 0A  INC (HL)
3E5E 3E 34
3E60 18 02 3E 35        487           ELSE             : DB 0A  DEC (HL)
3E64                    488           FI
3E64 CD 7B 5B           489           @[IY]A
3E67 CD 44 5B 03 6E 26 00 490         @[IY]SP DB 3  L=(HL) H=0
3E6E                    491 :
3E6E 18 34 FE 03 20 2A  492         EF A=INT$ THEN
3E74 CD A2 50           493           [TOP_ﾏｰｸ]
3E77 FE 01 20 05        494           IF A=定数% THEN
3E7B                    495 :           [POP_1] A=変数% [PUSH_A%HL]
3E7B CD 94 3D           496             [＋－変数処理]
3E7E 18 1C              497           ELSE
3E80 CD D6 4B           498             [ｺｰﾄﾞ生成1]
3E83 CD 44 5B 03 5E 23 56 499           @[IY]SP DB 3  E=(HL) INC HL D=(HL)
3E8A                    500 :           IF (＋－FLG)=＋$ : DB 0A  INC DE
3E8A                    501 :           ELSE             : DB 0A  DEC DE
3E8A                    502 :           FI
3E8A 3A BB 3F D6 10     503             A=(＋－FLG) SUB $10
3E8F CD 7B 5B           504             @[IY]A
3E92 CD 44 5B 03 72 2B 73 505           @[IY]SP DB 3  (HL)=D DEC HL (HL)=E
3E99 CD A5 4F           506             @[IY]EX
3E9C                    507           FI
3E9C                    508 :
3E9C 18 06              509         ELSE
3E9E CD E5 41 CD 94 3D  510           [配列参照処理] [＋－変数処理]
3EA4                    511         FI
3EA4 C9                 512         RET
3EA5                    513 :
3EA5                    514 :
3EA5 3E 03 11           515 [＋－PORTW]  A=INT$  DB SKIP
3EA8 3E 02              516 [＋－PORT]   A=CHR$
3EAA 32 48 41           517         (配列型)=A
3EAD                    518 :
3EAD CD F5 42           519         配列用[式]
3EB0                    520 :       [＋－PORT処理]
3EB0                    521 :       RET
3EB0                    522 :
3EB0                    523 [＋－PORT処理]
3EB0 CD 58 48           524         [ｺｰﾄﾞ生成1_POP]
3EB3                    525 :
3EB3 3A 48 41           526         A=(配列型)
3EB6 FE 02 20 22        527         IF A=CHR$ THEN
3EBA CD 44 5B 04 44 4D ED 528         @[IY]SP DB 4  BC=HL  IN L,(BC)
3EC1 68
3EC2 3A BB 3F FE 23 20 04 529         IF (＋－FLG)=＋$ : DB 0A  INC L
3EC9 3E 2C
3ECB 18 02 3E 2D        530           ELSE             : DB 0A  DEC L
3ECF                    531           FI
3ECF CD 7B 5B           532           @[IY]A
3ED2 CD 44 5B 04 ED 69 26 533         @[IY]SP DB 4  OUT (BC),L  H=0
3ED9 00
3EDA                    534 :
3EDA 18 15              535         ELSE
3EDC 21 71 00 CD 73 4F  536           HL=&[PORTW参照]-&R @[IY]call_RT
3EE2 3A BB 3F           537           A=(＋－FLG)
3EE5 CD 7B 5B           538           @[IY]A
3EE8 CD A5 4F           539           @[IY]EX
3EEB 21 82 00 CD 73 4F  540           HL=&[PORTW代入]-&R @[IY]call_RT
3EF1                    541         FI
3EF1                    542 :
3EF1 C3 BD 50           543         [PUSH_中間%] RET
3EF4                    544 :
3EF4                    545 :
3EF4 3E 03 11           546 [＋－SOSW]  A=INT$  DB SKIP
3EF7 3E 02              547 [＋－SOS]   A=CHR$
3EF9 32 48 41           548         (配列型)=A
3EFC                    549 :
3EFC CD F5 42           550         配列用[式]
3EFF                    551 :       [＋－SOS処理]
3EFF                    552 :       RET
3EFF                    553 :
3EFF                    554 [＋－SOS処理]
3EFF CD 58 48           555         [ｺｰﾄﾞ生成1_POP]
3F02                    556 :
3F02 3A 48 41           557         A=(配列型)
3F05 FE 02 20 22        558         IF A=CHR$ THEN
3F09 21 94 1F CD 71 5B  559           HL=#PEEK  @[IY]CALL
3F0F 3A BB 3F FE 23 20 04 560         IF (＋－FLG)=＋$ : DB 0A  INC A
3F16 3E 3C
3F18 18 02 3E 3D        561           ELSE             : DB 0A  DEC A
3F1C                    562           FI
3F1C CD 7B 5B           563           @[IY]A
3F1F CD 44 5B 06 CD 9A 1F 564         @[IY]SP DB 6  CALL #POKE  L=A H=0
3F26 6F 26 00
3F29                    565 :
3F29 18 15              566         ELSE
3F2B 21 4E 00 CD 73 4F  567           HL=&[SOSW参照]-&R @[IY]call_RT
3F31 3A BB 3F           568           A=(＋－FLG)
3F34 CD 7B 5B           569           @[IY]A
3F37 CD A5 4F           570           @[IY]EX
3F3A 21 5A 00 CD 73 4F  571           HL=&[SOSW代入]-&R @[IY]call_RT
3F40                    572         FI
3F40                    573 :
3F40 C3 BD 50           574         [PUSH_中間%] RET
3F43                    575 :
3F43                    576 :
3F43                    577 [PORT配＋－] :in A:＋$/－$
3F43 32 BB 3F           578         (＋－FLG)=A
3F46 CD B0 3E C3 5B 3F  579         [＋－PORT処理] [配列＋－後処理] RET
3F4C                    580 :
3F4C                    581 [SOS配＋－]  :in A:＋$/－$
3F4C 32 BB 3F           582         (＋－FLG)=A
3F4F CD FF 3E C3 5B 3F  583         [＋－SOS処理]  [配列＋－後処理] RET
3F55                    584 :
3F55                    585 [MEM配＋－]  :in A:＋$/－$
3F55 32 BB 3F           586         (＋－FLG)=A
3F58 CD 4D 3E           587         [＋－MEM処理]
3F5B                    588 :       [配列＋－後処理]
3F5B                    589 :       RET
3F5B                    590 :
3F5B                    591 [配列＋－後処理]
3F5B 3A 48 41           592         A=(配列型)
3F5E FE 02 20 12        593         IF A=CHR$ THEN
3F62 3A BB 3F FE 23 20 04 594         IF (＋－FLG)=＋$ : DB 0A  DEC L
3F69 3E 2D
3F6B 18 02 3E 2C        595           ELSE             : DB 0A  INC L
3F6F                    596           FI
3F6F CD 7B 5B           597           @[IY]A
3F72                    598 :
3F72 18 46 FE 03 20 12  599         EF A=INT$ THEN
3F78 3A BB 3F FE 23 20 04 600         IF (＋－FLG)=＋$ : DB 0A  DEC HL
3F7F 3E 2B
3F81 18 02 3E 23        601           ELSE             : DB 0A  INC HL
3F85                    602           FI
3F85 CD 7B 5B           603           @[IY]A
3F88                    604 :
3F88 18 30              605         ELSE
3F8A CD A2 50           606           [TOP_ﾏｰｸ]
3F8D FE 03 20 09        607           IF A=ｺｰﾄﾞ% THEN
3F91 CD DD 4F CD B4 50 CD 608           [TEMP代入処理] [POP_1] [PUSH_中間%]
3F98 BD 50
3F9A                    609           FI
3F9A 3E 03 32 6F 50 3E 01 610         (型)=INT$  A=定数%  HL=1 [PUSH_A%HL]
3FA1 21 01 00 CD BF 50
3FA7 CD 26 4C           611           [ｺｰﾄﾞ生成2]
3FAA 21 23 00           612           HL=#SUB
3FAD 3A BB 3F FE 2B 20 03 613         IF (＋－FLG)=－$ THEN HL=#ADD
3FB4 21 20 00
3FB7 CD 7D 4F           614           @[IY]call_SRB
3FBA                    615         FI
3FBA C9                 616         RET
3FBB                    617 :
3FBB 00                 618 ＋－FLG DB 0
3FBC                    619 :
3FBC                    620 :
3FBC                    621 [変数&記定]
3FBC CD 62 59           622         [局所表ｻｰﾁ]
3FBF D4 82 59           623         IF NC [大域表ｻｰﾁ]
3FC2 D4 16 40           624         IF NC [未宣言変数]
3FC5                    625 :
3FC5 F5 78 32 6F 50 F1  626         PUSH AF  (型)=B  POP AF
3FCB                    627 :
3FCB                    628 :[記号定数]
3FCB                    629 :
3FCB FE 01 20 0E        630         IF A=記定$ THEN
3FCF 78 FE 03 20 04 3E 01 631         IF B=INT$ : A=定数%
3FD6 18 02 3E 02        632           ELSE      : A=変数%
3FDA                    633           FI
3FDA C3 BF 50           634           [PUSH_A%HL] RET
3FDD                    635         FI
3FDD                    636 :
3FDD                    637 :[配列名]
3FDD                    638 :
3FDD FE 05 28 04 FE 06 20 639       IF A=配列1$ OR A=配列2$ THEN
3FE4 08
3FE5 3E 03 32 6F 50     640           (型)=INT$
3FEA C3 CE 4B           641           [PUSH_定数%HL]  RET
3FED                    642         FI
3FED                    643 :
3FED                    644 :[変数]
3FED                    645 :
3FED FE 03 28 04 FE 04 20 646       IF A=関変1$ OR A=関変2$ : (型)=INT$
3FF4 07 3E 03 32 6F 50
3FFA 18 07 FE 02 28 03 CD 647       EF A<>変数$            : [名前誤用]
4001 FB 61
4003                    648         FI
4003 3E 02 CD BF 50     649         A=変数% [PUSH_A%HL]
4008                    650 :
4008 CD 44 43           651         [代入等号＋－]
400B                    652 :参照
400B D0                 653         IF NC RET
400C                    654 :＋＋.－－
400C B7 C2 F3 3D        655         IF A<>0 [変数＋－] RET
4010                    656 :代入
4010 CD 70 3A C3 31 40  657         [論理項] [代入処理] RET
4016                    658 :
4016                    659 :
4016                    660 [未宣言変数]
4016 CD 99 62 EC 56 41 52 661       [ERROR] DM @UDEF,"VAR",0
401D 00
401E                    662 :
401E                    663 :       (表域)=大域$
401E 3E 03 32 52 5A     664         (登録型)=INT$
4023 3E 02 21 00 00 11 00 665       A=変数$  HL=0  DE=0  [名前登録]
402A 00 CD A7 36
402E                    666 :
402E 06 03              667         B=INT$
4030 C9                 668         RET
4031                    669 :
4031                    670 :
4031                    671 [代入処理]
4031 CD A7 50           672         [SEC_ﾏｰｸ]
4034                    673 :INT
4034 3A 6F 50 FE 03 20 12 674       IF (型)=INT$ THEN
403B CD D3 4B           675           [整数ｺｰﾄﾞ生成1]
403E CD B4 50           676           [POP_1]
4041 CD B4 50 3E 22 CD 73 677         [POP_1]  A=@[]HL  @[IY]AHL
4048 5B
4049 3E 03              678           A=INT$
404B                    679 :REAL
404B 18 1D              680         ELSE
404D CD 58 48 3A 6F 50  681           [ｺｰﾄﾞ生成1_POP] A=(型)
4053 F5                 682           PUSH AF
4054 CD B4 50 CD 5F 5B  683           [POP_1] @[IY]DEnn
405A F1                 684           POP  AF
405B 21 8C 00           685           HL=&MOVE_EX-&R
405E FE 03 20 03 21 95 00 686         IF A=INT$ THEN HL=&CVITF-&R
4065 CD 73 4F           687           @[IY]call_RT
4068 3E 04              688           A=REAL$
406A                    689         FI
406A 32 6F 50 C3 E3 3D  690         (型)=A  [PUSH_ｺｰﾄﾞ%]  RET
4070                    691 :
4070                    692 :
4070                    693 [配列]
4070 CD 67 60           694         [TBLｻｰﾁ]
4073 4D 45 CD 6D 41     695         DM "MEM"+$80    DW [MEM配列]
4078 4D 45 4D D7 70 41  696         DM "MEMW"+$80   DW [MEMW配列]
407E 4D 45 4D D2 73 41  697         DM "MEMR"+$80   DW [MEMR配列]
4084 50 4F 52 D4 42 42  698         DM "PORT"+$80   DW [PORT配列]
408A 50 4F 52 54 D7 3F 42 699       DM "PORTW"+$80  DW [PORTW配列]
4091 53 4F D3 6A 42     700         DM "SOS"+$80    DW [SOS配列]
4096 53 4F 53 D7 67 42  701         DM "SOSW"+$80   DW [SOSW配列]
409C 43 4F 4E 53 D4 C1 42 702       DM "CONST"+$80  DW [CONST配列]
40A3 43 4F 4E 53 54 D7 C4 703       DM "CONSTW"+$80 DW [CONSTW配列]
40AA 42
40AB 00                 704         DM 0
40AC DA 81 1F           705         IF C CALL [HL]  RET
40AF                    706 :
40AF CD BB 40           707         [配列処理]
40B2                    708 :
40B2 3A 49 41 FE 01 CC 7B 709       IF (代入FLG)=TRUE [配列代入参照]
40B9 41
40BA C9                 710         RET
40BB                    711 :
40BB                    712 :
40BB                    713 [配列処理]
40BB CD 62 59           714         [局所表ｻｰﾁ]
40BE D4 82 59           715         IF NC [大域表ｻｰﾁ]
40C1 D4 4A 41           716         IF NC [未宣言配列]
40C4                    717 :       A:type  B:型  HL:data  (補助DATA)
40C4 4F                 718         C=A
40C5 F5                 719         PUSH AF
40C6 78 32 48 41 3E 03 32 720       (配列型)=B  (型)=INT$
40CD 6F 50
40CF                    721 :
40CF 79                 722         A=C
40D0 FE 05 28 04 FE 06 20 723       IF A=配列1$ OR A=配列2$ : A=定数%
40D7 04 3E 01
40DA 18 11 FE 03 28 04 FE 724       EF A=関変1$ OR A=関変2$ : A=変数%
40E1 04 20 04 3E 02
40E6 18 05 CD FB 61 3E 01 725       ELSE          [名前誤用]  A=定数%
40ED                    726         FI
40ED CD BF 50           727         [PUSH_A%HL]
40F0                    728 :
40F0 CD F5 42           729         配列用[式]
40F3                    730 :
40F3 3E 01 32 49 41     731         (代入FLG)=TRUE
40F8                    732 :
40F8                    733 :2次元
40F8 F1                 734         POP  AF
40F9 FE 06 28 04 FE 04 20 735       IF A=配列2$ OR A=関変2$ THEN
4100 22
4101 3E 03 32 6F 50     736           (型)=INT$
4106 2A 55 5A           737           HL=(補助DATA)
4109 CD CE 4B CD 10 49  738           [PUSH_定数%HL] [乗算処理]
410F                    739 :
410F DD 7E 00 FE 5B 20 08 740         IF (IX)="[" THEN
4116 CD F5 42 CD F8 48  741             配列用[式] [加算処理]
411C 18 05              742           ELSE
```

▶ド素人の私にとっては6月号は読みがいがあった。Oh!Xに載っている内容は私にとって難しいのでこういう特集はうれしい。　古味　貴徳(17)千葉県

```
411E 3E 00 32 49 41      743                (代入FLG)=FALSE
4123                     744              FI
4123                     745            FI
4123                     746 :
4123 26 00 3A AE 69 6F   747            H=0  L=(精度)
4129 3A 48 41            748            A=(配列型)
412C FE 02 20 04 2E 01   749            IF A=CHR$ : L=1
4132 18 06 FE 03 20 02 2E 750           EF A=INT$ : L=2
4139 02
413A                     751 :          ELSE      : L=(精度)
413A                     752            FI
413A 3E 03 32 6F 50 CD CE 753           (型)=INT$ [PUSH_定数%HL] [乗算処理]
4141 4B CD 10 49
4145                     754 :
4145 C3 FB 48            755            [加算処理] RET
4148                     756 :
4148 00                  757 配列型  DB 0
4149 00                  758 代入FLG DB 0
414A                     759 :
414A                     760 :
414A                     761 [未宣言配列]
414A CD 99 62 EC 41 52 52 762           [ERROR] DM @UDEF,"ARRAY",0
4151 41 59 00
4154                     763 :
4154                     764 :          (表域)=大域$
4154 3E 03 32 52 5A      765            (登録型)=INT$
4159 3E 05 21 00 00 11 00 766           A=配列1$  HL=0  DE=0  [名前登録]
4160 00 CD A7 36
4164                     767 :
4164 06 03 AF 32 55 5A 3E 768           B=INT$ (補助DATA)=0  A=配列1$
416B 05
416C C9                  769            RET
416D                     770 :
416D                     771 :
416D 3E 02 11            772 [MEM配列]  A=CHR$  DB SKIP
4170 3E 03 11            773 [MEMW配列] A=INT$  DB SKIP
4173 3E 04               774 [MEMR配列] A=REAL$
4175 32 48 41            775            (配列型)=A
4178                     776 :
4178 CD F5 42            777            配列用[式]
417B                     778 :
417B                     779 :          [配列代入参照]
417B                     780 :          RET
417B                     781 :
417B                     782 [配列代入参照]  ;in (配列型)
417B CD 44 43            783            [代入等号+-]
417E                     784 ;参照
417E D2 E5 41            785            IF NC [配列参照処理] RET
4181                     786 ;++,--
4181 B7 C2 55 3F         787            IF A<>0 [MEM配+-] RET
4185                     788 ;代入
4185 3A 48 41 F5 CD 70 3A 789           A=(配列型)  PUSH AF  [論理項]  POP AF
418C F1
418D 32 48 41            790            (配列型)=A
4190                     791 :
4190 FE 02 20 12         792            IF A=CHR$ THEN
4194 CD 2E 4D CD 26 4C CD 793             [TOP整数化] [ｺｰﾄﾞ生成2] [POP_1]
419B B4 50
419D CD 3F 5B 73         794              @[IY]SP1 : (HL)=E
41A1 CD A5 4F            795              @[IY]EX
41A4                     796 :
41A4 18 39 FE 03 20 15   797            EF A=INT$ THEN
41AA CD 2E 4D CD 26 4C CD 798             [TOP整数化] [ｺｰﾄﾞ生成2] [POP_1]
41B1 B4 50
41B3 CD 44 5B 03 73 23 72 799             @[IY]SP  DB 3 (HL)=E  INC HL  (HL)=D
41BA CD A5 4F            800              @[IY]EX
41BD                     801 :
41BD 18 20               802            ELSE
41BF CD A2 50 3A 6F 50   803              [TOP_ﾏｰｸ]  A=(型)
41C5 F5                  804              PUSH AF
41C6 CD 53 4C CD B4 50   805              [無変換ｺｰﾄﾞ生成2] [POP_1]
41CC F1                  806              POP  AF
41CD FE 03 20 08         807              IF A=INT$ THEN
41D1 21 14 00 CD 7D 4F   808                HL=#CVITF    @[IY]call_SRB
41D7 18 06               809              ELSE
41D9 21 8B 00 CD 73 4F   810                HL=&MOVE_R-&R @[IY]call_RT
41DF                     811              FI
41DF                     812            FI
41DF                     813 :
41DF CD E6 42 C3 E3 3D   814            [SET配列PUSH型] [PUSH_ｺｰﾄﾞ%] RET
41E5                     815 :
41E5                     816 :
41E5                     817 [配列参照処理]
41E5 3A 48 41            818            A=(配列型)
41E8 FE 02 20 11         819            IF A=CHR$ THEN
41EC CD D3 4B CD B4 50   820              [整数ｺｰﾄﾞ生成1] [POP_1]
41F2 CD 44 5B 03 6E 26 00 821             @[IY]SP DB 3  L=(HL) H=0
41F9 3E 00               822              A=中間%
41FB                     823 :
41FB 18 3C FE 03 20 22   824            EF A=INT$ THEN
4201 CD A2 50            825              [TOP_ﾏｰｸ]
4204 FE 01 20 07         826              IF A=定数% THEN
4208 CD B4 50            827                [POP_1]
420B 3E 02               828                A=変数%
420D 18 12               829              ELSE
420F CD D3 4B CD B4 50   830                [整数ｺｰﾄﾞ生成1] [POP_1]
4215 CD 44 5B 03 5E 23 56 831               @[IY]SP DB 3  E=(HL) INC HL D=(HL)
421C CD A5 4F            832                @[IY]EX
421F 3E 00               833                A=中間%
4221                     834              FI
4221                     835 :
4221 18 16               836            ELSE
4223 CD A2 50            837              [TOP_ﾏｰｸ]
4226 FE 01 20 07         838              IF A=定数% THEN
422A CD B4 50            839                [POP_1]
422D 3E 02               840                A=変数%
422F 18 08               841              ELSE
4231 CD D3 4B CD B4 50   842                [整数ｺｰﾄﾞ生成1] [POP_1]
4237 3E 03               843                A=ｺｰﾄﾞ%
4239                     844              FI
4239                     845            FI
4239                     846 :          PUSH AF
4239 CD E6 42            847            [SET配列PUSH型]
423C                     848 :          POP  AF
423C C3 BF 50            849            [PUSH_A%HL] RET
423F                     850 :
423F                     851 :
423F 3E 03 11            852 [PORTW配列]  A=INT$  DB SKIP
4242 3E 02               853 [PORT配列]   A=CHR$
4244 32 48 41            854            (配列型)=A
4247                     855 :
4247 CD F5 42            856            配列用[式]
424A                     857 :
424A CD 44 43            858            [代入等号+-]
424D                     859 ;参照
424D 38 06               860            IF NC THEN
424F 21 5F 42 C3 8F 42   861              HL=PORT参照TBL [SYS配列参照処理] RET
4255                     862            FI
4255                     863 ;+-
4255 B7 C2 43 3F         864            IF A<>0 [PORT配+-] RET
4259                     865 ;代入
4259 21 63 42 C3 98 42   866            HL=PORT代入TBL [SYS配列代入処理] RET
425F                     867 :
425F                     868 PORT参照TBL
425F 68 00               869            DW &[PORT参照]-&R
4261 71 00               870            DW &[PORTW参照]-&R
4263                     871 :
4263                     872 PORT代入TBL
4263 7C 00               873            DW &[PORT代入]-&R
4265 82 00               874            DW &[PORTW代入]-&R
4267                     875 :
4267                     876 :
4267 3E 03 11            877 [SOSW配列]  A=INT$  DB SKIP
426A 3E 02               878 [SOS配列]   A=CHR$
426C 32 48 41            879            (配列型)=A
426F                     880 :
426F CD F5 42            881            配列用[式]
4272                     882 :
4272 CD 44 43            883            [代入等号+-]
4275                     884 ;参照
4275 38 06               885            IF NC THEN
4277 21 87 42 C3 8F 42   886              HL=SOS参照TBL [SYS配列参照処理] RET
427D                     887            FI
427D                     888 ;+-
427D B7 C2 4C 3F         889            IF A<>0 [SOS配+-] RET
4281                     890 ;代入
4281 21 8B 42 C3 98 42   891            HL=SOS代入TBL [SYS配列代入処理] RET
4287                     892 :
4287                     893 SOS参照TBL
4287 45 00               894            DW &[SOS参照]-&R
4289 4E 00               895            DW &[SOSW参照]-&R
428B                     896 :
428B                     897 SOS代入TBL
428B 52 00               898            DW &[SOS代入]-&R
428D 5A 00               899            DW &[SOSW代入]-&R
428F                     900 :
428F                     901 :
428F                     902 [SYS配列参照処理] ;in HL:TBL
428F E5                  903            PUSH HL
4290                     904 :
4290 CD D3 4B            905            [整数ｺｰﾄﾞ生成1]
4293                     906 :
4293 3A 48 41 18 12      907            A=(配列型)  JR [SYS配代処理]1
4298                     908 :
4298                     909 [SYS配列代入処理] ;in HL:TBL
4298 E5                  910            PUSH HL
4299                     911 :
4299 3A 48 41            912            A=(配列型)
429C F5                  913            PUSH AF
429D CD 70 3A CD 2E 4D CD 914           [論理項] [TOP整数化] [ｺｰﾄﾞ生成2]
42A4 26 4C
42A5 F1                  915            POP AF
42A7 32 48 41            916            (配列型)=A
42AA                     917 [SYS配代処理]1
42AA E1 FE 03 20 02 23 23 918           POP HL IF A=INT$ THEN  INC HL/HL
42B1 7E 23               919            A=(HL) INC HL
42B3 66 6F CD 73 4F      920            H=(HL) L=A      @[IY]call_RT
42B8                     921 :
42B8 CD B4 50            922            [POP_1]
42BB CD E6 42 C3 E3 3D   923            [SET配列PUSH型] [PUSH_ｺｰﾄﾞ%] RET
42C1                     924 :
42C1                     925 :
42C1 3E 02 11            926 [CONST配列]  A=CHR$  DB SKIP
42C4 3E 03               927 [CONSTW配列] A=INT$
42C6 F5                  928            PUSH AF
42C7                     929 :
42C7 CD 1D 60 5B         930            [SPINC] DB "["
42CB                     931 :
42CB CD 22 3A            932            [INT定数式]
42CE 7E 23               933            A=(HL) INC HL
42D0 66                  934            H=(HL)
42D1 6F                  935            L=A
42D2 F1 FE 02 20 02 26 00 936           POP AF  IF A=CHR$ THEN H=0
42D9                     937 :
42D9 3E 03 32 6F 50 CD CE 938           (型)=INT$  [PUSH_定数%HL]
42E0 48
42E1                     939 :
42E1 3E 5D C3 03 60      940            A="]" [INCIX] RET
42E6                     941 :
42E6                     942 :
42E6                     943 [SET配列PUSH型]
42E6 F5                  944            PUSH AF
42E7                     945 :
42E7 3A 48 41 FE 02 20 02 946           A=(配列型) IF A=CHR$ THEN A=INT$
42EE 3E 03
42F0 32 6F 50            947            (型)=A
42F3                     948 :
42F3 F1                  949            POP  AF
42F4 C9                  950            RET
42F5                     951 :
42F5                     952 :
42F5                     953 配列用[式]
42F5 3A 48 41 47 3A 49 41 954           B=(配列型)  C=(代入FLG)  D=(+-FLG)
42FC 4F 3A BB 3F 57
4301 2A 55 5A            955            HL=(補助DATA)
4304 C5 D5 E5            956            PUSH BC/DE/HL
4307                     957 :
4307 CD 1D 60 5B         958            [SPINC] DB "["
430B DD 7E 00 FE 5D 20 0D 959           IF (IX)="]" THEN
4312 3E 03 32 6F 50 21 00 960             (型)=INT$  HL=0  [PUSH_定数%HL]
4319 00 CD CE 48
431D 18 0E               961            ELSE
431F CD 54 3A            962              [論理式]
4322 CD A2 50 3A 6F 50 FE 963             [TOP_ﾏｰｸ] IF (型)=REAL$ [REAL不可]
4329 04 CC 08 62
432D                     964            FI
432D CD 1D 60 5D         965            [SPINC] DB "]"
4331                     966 :
4331 E1 D1 C1            967            POP  HL/DE/BC
4334 78 32 48 41 79 32 49 968           (配列型)=B  (代入FLG)=C  (+-FLG)=D
433B 41 7A 32 BB 3F
4340 22 55 5A            969            (補助DATA)=HL
4343 C9                  970            RET
4344                     971 :
4344                     972 :
4344                     973 [代入等号+-]
4344 CD 67 60            974            [TBLｻｰﾁ]
4347 2B 2B 0D 23 00      975            DM "++",$0D DW +$
434C 2D 2D 0D 23 00      976            DM "--",$0D DW -$
4351 00                  977            DM 0
4352 7D D8               978            A=L IF C RET
4354                     979 :
4354 DD 7E 00 FE 3D 20 0C 980           IF (IX)="=" AND (IX+1)<>"=" THEN
435B DD 7E 01 FE 3D 28 05
4362 DD 23               981              INC IX
4364 AF 37 C9            982              A=0 SCF RET
4367                     983            FI
4367                     984 :
4367 AF B7 C9            985            A=0 RCF RET
436A                     986 :
436A                     987 :
436A                     988 [定数ｽﾀｯｸ初期化]
436A 21 61 44 22 5F 44   989            (定数ﾎﾟｲﾝﾀ)=HL=定数ｽﾀｯｸ
4370 C9                  990            RET
4371                     991 :
4371                     992 :
4371                     993 [定数]
4371 3E 03 32 6F 50      994            (型)=INT$
4376                     995 :
4376 DD 7E 00            996            A=(IX)
4379 FE 24 20 29         997            IF A="$" THEN
437D DD 23               998              INC IX
437F DD 7E 00 CD DD 60 CD 999             A=(IX) [TOUPPER] CALL #HEX
4386 B8 1F
4388                     1000 ;ｵﾌﾞｼﾞｪｸﾄ生成ｱﾄﾞﾚｽを値とする
4388 30 04 FD E5 E1 C9   1001             IF C THEN  HL=IY  RET ;SCF
438E                     1002 ;16進数          例 $ABCD
438E 21 00 00 54         1003             HL=0  D=H ;D=0
4392                     1004             {
4392 DD 7E 00            1005               A=(IX)
4395 CD DD 60 CD 38 1F D8 1006              [TOUPPER] CALL #HEX  IF C RET ;SCF
439C DD 23               1007               INC IX
439E 5F                  1008               E=A
439F 29 29               1009               ADD HL,HL  ADD HL,HL
43A1 29 29               1010               ADD HL,HL  ADD HL,HL
43A3 19                  1011               ADD HL,DE
43A4 18 EC               1012             }
43A6                     1013           FI
43A6                     1014 :
43A6                     1015 ;文字定数        例 'A','漢'
43A6 FE 27 20 1C         1016           IF A="'" THEN
43AA DD 23               1017             INC IX
43AC 21 00 00 CD 90 45   1018             HL=0 [文字定数処理]
43B2                     1019 :
43B2 DD 7E 00 FE 27 28 04 1020            IF (IX)<>"'" THEN H=L [文字定数処理]
43B9 65 CD 90 45
43BD                     1021 :
43BD E5 3E 27 CD 08 60 E1 1022            PUSH HL  A="'" [NOSPC_INCIX]  POP HL
43C4 37                  1023             SCF
43C5 C9                  1024             RET
43C6                     1025           FI
43C6                     1026 :
43C6                     1027 :         A=(IX)
43C6 CD EA 60 D0         1028           [数字ﾁｪｯｸ] IF NC RET
43CA                     1029 :
43CA DD E5               1030           PUSH IX
43CC DD 7E 00 DD 23 CD EA 1031          { A=(IX) INC IX  [数字ﾁｪｯｸ] } UNTIL NC
43D3 60 38 F6
43D5                     1032 :         A=(IX)
43D5 DD E1               1033           POP  IX
43D8                     1034 :
43D8                     1035 ;2進数           例 10101010B
43D8 CD DD 60            1036           [TOUPPER]
43DB FE 42 20 20         1037           IF A="B" THEN
43DF 21 00 00 55         1038             HL=0  D=L ;D=0
43E3                     1039             {
43E3 DD 7E 00            1040               A=(IX)
43E5 FE 30 28 04 FE 31 20 1041              IF A<>"0" THEN IF A<>"1" EXIT
43ED 09
43EE DD 23               1042               INC IX
43F0 D6 30 5F            1043               SUB A,"0" E=A
43F3                     1044 :
43F3 29                  1045               ADD HL,HL
43F4 19                  1046               ADD HL,DE
43F5 18 EC               1047             }
43F7 E5 CD 1D 60 42 E1   1048             PUSH HL  [SPINC] DB "B"  POP HL
43FD 37                  1049             SCF
43FE C9                  1050             RET
43FF                     1051           FI
43FF                     1052 :
43FF                     1053 ;REAL型として読む(正の数)
43FF                     1054 :
43FF F5                  1055           PUSH AF
4400 3A AE 69 CD 36 44   1056           A=(精度) [INC定数ﾎﾟｲﾝﾀ]
4406                     1057 :         HL=(定数ﾎﾟｲﾝﾀ)
4406 DD E5 D1 01 0E 00 CD 1058          DE=IX  BC=#CVSTF  CALL[SRB]  IX=DE
440D 87 4F D5 DD E1
4412 F1                  1059           POP  AF
4413                     1060 :
4413                     1061 ;INT型におさまればINT型にする
4413                     1062 :
4413                     1063 :         [TOUPPER]
4413 FE 45 28 18 FE 2E 28 1064          IF A<>"E" AND A<>"." THEN
441A 14
441B 7E FE 91 30 0F      1065             IF (HL)<=$90 THEN
4420 3A AE 69 CD 4E 44   1066               A=(精度) [DEC定数ﾎﾟｲﾝﾀ]
4426 01 1A 00 CD 87 4F E3 1067              BC=#CVFTU  CALL[SRB]  EX DE,HL
442D 37                  1068               SCF
442E C9                  1069               RET
442F                     1070             FI
442F                     1071           FI
442F                     1072 :         HL=実数値のｱﾄﾞﾚｽ
442F 3E 04 32 6F 50      1073           (型)=REAL$
4434 37                  1074           SCF
4435 C9                  1075           RET
4436                     1076 :
4436                     1077 :
4436                     1078 [INC定数ﾎﾟｲﾝﾀ]  ;in A(ﾊﾞｲﾄ)
4436 2A 5F 44            1079           HL=(定数ﾎﾟｲﾝﾀ)
4439 E5                  1080           PUSH HL
443A 85 6F               1081           ADD L,A
443C 7C CE 00 67         1082           ADC H,0
4440 22 5F 44            1083           (定数ﾎﾟｲﾝﾀ)=HL
4443 E1                  1084           POP  HL
4444 7D D6 A1 7C DE 44 D2 1085          IF HL>=定数ｽﾀｯｸEND [定数ｽﾀｯｸOVER!] RET
444B 9B 61
444D C9                  1086           RET
444E                     1087 :
444E                     1088 [DEC定数ﾎﾟｲﾝﾀ]  ;in A(ﾊﾞｲﾄ)
444E E5 D5               1089           PUSH HL/DE
4450 2A 5F 44            1090           HL=(定数ﾎﾟｲﾝﾀ)
4453 5F                  1091           E=A
4454 16 00               1092           D=0
4456 B7 ED 52            1093           SUB HL,DE
4459 22 5F 44            1094           (定数ﾎﾟｲﾝﾀ)=HL
445C D1 E1               1095           POP DE/HL
445E C9                  1096           RET
445F                     1097 :
445F 00 00               1098 定数ﾎﾟｲﾝﾀ   DW $0000
4461 00 00 00 00 00 00 00 1099 定数ｽﾀｯｸ    DS 定数ｽﾀｯｸ数*8
4468 00 00 00 00 00 00 00
446F 00 00 00 00 00 00 00
4476 00 00 00 00 00 00 00
447D 00 00 00 00 00 00 00
4484 00 00 00 00 00 00 00
448B 00 00 00 00 00 00 00
4492 00 00 00 00 00 00 00
4499 00 00 00 00 00 00 00
44A0 00
44A1                     1100 定数ｽﾀｯｸEND
44A1                     1101 :
44A1                     1102 :
44A1 CD AD 44 C3 6D 5B   1103 @HL確定  [確定定数処理]  @[IY]HLnn  RET
44A7 CD AD 44 C3 5F 5B   1104 @DE確定  [確定定数処理]  @[IY]DEnn  RET
44AD                     1105 :
44AD                     1106 [確定定数処理] ;out HL
44AD CD D0 50 FD E5 D1 13 1107          [POP_DATA]  DE=IY  INC DE
44B4                     1108 :
44B4 22 65 45            1109           (確定定数BUFF)=HL
44B7 ED 53 67 45         1110           (確定定数DATA)=DE
44BB 3A AE 69 32 69 45   1111           (確定定数BYTE)=(精度)
44C1                     1112 :
44C1 2A 8A 45            1113           HL=(定数表TOP)
44C4                     1114           {
44C4 CD 94 1F B7 28 3F   1115             #PEEK  IF A=0 EXIT
44CA                     1116 :
44CA 47 23 23 23         1117             B=A  INC HL/HL/HL
44CE                     1118 :
44CE 3A 69 45 B8 20 30   1119             IF (確定定数BYTE)=B THEN
44D4 E5                  1120               PUSH HL
44D5 ED 5B 65 45         1121               DE=(確定定数BUFF)
44D9                     1122               DO B {
44D9 CD 94 1F 4F 1A B9 20 1123               #PEEK  C=A  IF (DE)<>C EXIT
44E0 04
44E1 23 13               1124                 INC HL/DE
44E3 10 F4               1125               }
44E5 D1                  1126               POP DE
44E6                     1127 :             ﾃﾞｰﾀが一致したら
44E6 04 05 20 1A         1128               IF B=0 THEN
44EA EB ED 5B 67 45      1129                 EX DE,HL  DE=(確定定数DATA)
44EF 2B CD 94 1F 47 7A CD 1130                DEC HL  #PEEK B=A  A=D #POKE
44F6 9A 1F
44F8 2B CD 94 1F 4F 7B CD 1131                DEC HL  #PEEK C=A  A=E #POKE
44FF 9A 1F
4501 60 69               1132                 HL=BC
4503 C9                  1133                 RET
4504                     1134               FI
4504                     1135             FI
4504                     1136 :
4504 23 10 FD            1137             DO B { INC HL }
4507 18 BB               1138           }
4509 2A 8E 45            1139           HL=(定数表ﾎﾟｲﾝﾀ)
450C ED 5B 67 45         1140           DE=(確定定数DATA)
4510 3A 69 45            1141           A=(確定定数BYTE)
4513 CD 38 45            1142               [定数表POKE] ;ﾊﾞｲﾄ数
4516 7B CD 38 45         1143           A=E  [定数表POKE] ;ﾁｪｲﾝ用
451A 7A CD 38 45         1144           A=D  [定数表POKE] ;ﾁｪｲﾝ用
451E ED 5B 65 45         1145           DE=(確定定数BUFF)
4522 3A 69 45 47         1146           DO B,(確定定数BYTE) {
4526 1A CD 38 45 13      1147             A=(DE) [定数表POKE]  INC DE
452B 10 F9               1148           }
452D 22 8E 45            1149           (定数表ﾎﾟｲﾝﾀ)=HL
4530                     1150 :
4530 AF CD 38 45         1151           A=0 [定数表POKE]
4534                     1152 :
4534 21 00 00            1153           HL=0
4537 C9                  1154           RET
4538                     1155 :
4538                     1156 :
4538                     1157 [定数表POKE] ;in HL,A
4538 F5 C5 D5            1158           PUSH AF/BC/DE
453B                     1159 :
453B ED 5B 8C 45         1160           DE=(定数表END)
453F ED 4B 65 45 79 C6 04 1161          BC=(確定定数BUFF) ADD BC,4
4546 4F 78 CE 00 47
454B 7B 91 5F 7A 98 57   1162           SUB DE,BC
4551 7D 93 7C 9A D4 75 61 1163          IF HL>=DE [定数表OVER!]
4558                     1164 :
4558 D1 C1 F1            1165           POP DE/BC/AF
455B                     1166 :         #POKE+
455B                     1167 :         RET
455B                     1168 :
455B                     1169 #POKE+
455B CD 9A 1F 23 C9      1170           #POKE  INC HL  RET
4560                     1171 :
4560                     1172 #PEEK+
4560 CD 94 1F 23 C9      1173           #PEEK  INC HL  RET
4565                     1174 :
4565 00 00               1175 確定定数BUFF  DW $0000
4567 00 00               1176 確定定数DATA  DW $0000
4569 00                  1177 確定定数BYTE  DB 0
456A                     1178 :
456A                     1179 :
456A                     1180 [定数ﾁｪｲﾝ]
456A 2A 8A 45            1181           HL=(定数表TOP)
456D                     1182           {
456D CD 60 45 B7 C8      1183             #PEEK+  IF A=0 RET
4572 F5                  1184             PUSH AF
4573 CD 60 45 5F         1185             #PEEK+  E=A
4577 CD 60 45 57         1186             #PEEK+  D=A
457B                     1187 :           PUSH HL
457B CD 41 50            1188             [ﾁｪｲﾝ処理]
457E                     1189 :           POP  HL
457E F1                  1190             POP  AF
457F 47 CD 60 45 CD 7B 5B 1191            DO B,A {  #PEEK+  @[IY]A }
4586 10 F8
4588 18 E3               1192           }
458A                     1193 :
458A 00 00               1194 定数表TOP    DW $0000
458C 00 00               1195 定数表END    DW $0000
458E 00 00               1196 定数表ﾎﾟｲﾝﾀ DW $0000
4590                     1197 :
4590                     1198 :
4590                     1199 [文字定数処理]
4590 DD 7E 00            1200           A=(IX)
4593 FE 20 30 05         1201           IF A<$20 THEN
4597 CD E4 61            1202             [文字列ｴﾗｰ]
459A 18 07               1203           ELSE
459C DD 23 FE 5C CC 42 45 1204            INC IX  IF A="¥" [¥文字]
45A3                     1205           FI
45A3 6F C9               1206           L=A  RET
45A5                     1207 :
45A5                     1208 :
45A5                     1209 [文字列]
45A5 CD 91 4F            1210           @PUSH_HL
45A8 3E 03 32 6F 50 CD ED 1211          (型)=INT$ [PUSH_中間%]
45AF 50
45B0                     1212 :         [文字列処理]
45B0                     1213 :         RET
45B0                     1214 :
45B0                     1215 [文字列処理]  ;PRINT関数でも使用
45B0 21 00 00 CD 6D 5B   1216           HL=0  @[IY]HLnn
45B6                     1217 :
45B6 FD E5 D1 1B 1B      1218           DE=IY  DEC DE/DE
45BB 2A 8E 45            1219           HL=(定数表ﾎﾟｲﾝﾀ)
45BE 23                  1220           INC HL          ;ﾊﾞｲﾄ数
45BF 7B CD 38 45         1221           A=E [定数表POKE]  ;ﾁｪｲﾝ用
45C3 7A CD 38 45         1222           A=D [定数表POKE]  ;ﾁｪｲﾝ用
45C7 AF 32 42 61 06 01   1223           (漢ﾌﾗｸﾞ)=0  B=1
45CD                     1224           {
45CD DD 7E 00            1225             A=(IX)
45D0 FE 20 30 09         1226             IF A<$20 THEN
45D4 C5 E5 CD E4 61 E1 C1 1227              PUSH BC/HL  [文字列ｴﾗｰ]  POP HL/BC
45DB 18 1D               1228               EXIT
45DD                     1229             FI
```

▶おおう。5月5日で20歳になってしまった。ううっ，悲しい。もう青春は帰ってこない……。

井上　敬介(20)神奈川県

```
45DD DD 23              1230            INC IX
45DF FE 22 28 17        1231            IF A="""" EXIT
45E3                    1232 :
45E3 CD 0F 46 CD 38 45  1233            [文字列漢字#処理] [定数表POKE]
45E9                    1234 :
45E9 04                 1235            INC B
45EA 78 FE FF 20 09     1236            IF B=255 THEN
45EF C5 E5 CD EB 61 E1 C1 1237            PUSH BC/HL  [LONG文字列] POP HL/BC
45F6 18 02              1238              EXIT
45F8                    1239            FI
45F8 18 D3              1240          }
45FA AF CD 38 45        1241          A=0 [定数表POKE]          :EOS
45FE AF CD 38 45 2B     1242          A=0 [定数表POKE] DEC HL
4603 ED 5B 8E 45 22 8E 45 1243        DE=(定数表ﾎﾟｲﾝﾀ)  (定数表ﾎﾟｲﾝﾀ)=HL
460A EB                 1244          EX DE,HL
460B 78 C3 9A 1F        1245          A=B #POKE  RET  :ﾊﾞｲﾄ数
460F                    1246 :
460F                    1247 :
460F                    1248 [文字列漢字#処理] :CODE関数でも使用
460F                    1249 ;in A,(漢ﾌﾗｸﾞ)  out A
460F 4F                 1250          C=A
4610 3A 42 61 FE 01 20 05 1251        IF (漢ﾌﾗｸﾞ)=1 THEN
4617 AF 32 42 61        1252            (漢ﾌﾗｸﾞ)=0
461B 18 23              1253          ELSE
461D 79                 1254            A=C
461E FE 80 38 0B FE A0 30 1255          IF A>=$80 AND A<$A0 : (漢ﾌﾗｸﾞ)=1
4625 07 3E 01 32 42 61
4628 18 13 FE E0 38 07 3E 1256          EF A>=$E0            : (漢ﾌﾗｸﾞ)=1
4632 01 32 42 61
4636 18 08 FE 5C 20 04 CD 1257          EF A="\"            : [\文字]  C=A
463D 42 46 4F
4640                    1258            FI
4640                    1259          FI
4640 79                 1260          A=C
4641 C9                 1261          RET
4642                    1262 :
4642                    1263 :
4642                    1264 [\文字]
4642 C5 D5 E5           1265          PUSH BC/DE/HL
4645                    1266 :        \$0D
4645 DD 7E 00 FE 24 20 0F 1267        IF (IX)="$" THEN
464C DD E5 D1           1268            DE=IX
464F 13                 1269            INC DE
4650 CD B5 1F D5 DD E1 DC 1270          CALL #AHEX  IX=DE  IF C [文字列ｴﾗｰ]
4657 E4 61
4659                    1271 :        \n
4659 18 31              1272          ELSE
465B CD 6A 60           1273            [TBLｻｰﾁ2]
465E 5C 0D 5C 00        1274            DM "\",$0D  DW "\"
4662 22 0D 22 00        1275            DM """",$0D  DW """"
4666 27 0D 27 00        1276            DM "'",$0D  DW "'"
466A 4E 0D 0D 00        1277            DM "N",$0D  DW $0D
466E 43 0D 0C 00        1278            DM "C",$0D  DW $0C
4672 52 0D 1C 00        1279            DM "R",$0D  DW $1C
4676 4C 0D 1D 00        1280            DM "L",$0D  DW $1D
467A 55 0D 1E 00        1281            DM "U",$0D  DW $1E
467E 44 0D 1F 00        1282            DM "D",$0D  DW $1F
4682 30 0D 00 00        1283            DM "0",$0D  DW $00
4686 00                 1284            DM 0
4687 38 02 2E 5C        1285            IF NC THEN  L="\"
468B 7D                 1286            A=L
468C                    1287          FI
468C                    1288 :
468C E1 D1 C1           1289          POP HL/DE/BC
468F C9                 1290          RET
4690                    1291 :
4690                    1292 :
4690                    1293 [関数ｺｰﾙ]
4690 CD 91 4F           1294          @PUSH_HL
4693                    1295 :
4693 CD 67 60           1296          [TBLｻｰﾁ]
4696 43 4F 44 C5 EB 56  1297          DM "CODE"+$80  DW [CODE関数]
469C 50 52 49 4E D4 80 57 1298        DM "PRINT"+$80 DW [PRINT関数]
46A3 50 55 53 C8 A0 56  1299          DM "PUSH"+$80  DW [PUSH関数]
46A9 50 4F D0 B8 56     1300          DM "POP"+$80   DW [POP関数]
46AE 00                 1301          DM 0
46AF 30 13              1302          IF C THEN
46B1 E5                 1303            PUSH HL
46B2 CD 38 58           1304            [ｶｯｺ開き]
46B5 E1 CD 81 1F        1305            POP  HL  CALL [HL]
46B9 CD 36 58           1306            [ｶｯｺ閉じ]
46BC 3E 03 32 6F 50 C3 E3 1307          (型)=INT$ [PUSH_ｺｰﾄﾞ%] RET
46C3 3D
46C4                    1308          FI
46C4                    1309 :
46C4                    1310 :        [名前ﾁｪｯｸ]
46C4 CD 82 59 4F 78 32 6F 1311        [大域表ｻｰﾁ] C=A  (型)=B
46CB 50
46CC 38 0F              1312          IF NC THEN
46CE                    1313 :          (表域)=大域$
46CE 3E 03 32 52 5A CD 18 1314          (登録型)=INT$ [関数名登録]
46D5 38
46D6 3E 03 32 6F 50 0E 00 1315          (型)=INT$ C=0
46DD                    1316          FI
46DD                    1317 :
46DD 3A 6F 50 F5        1318          A=(型)       PUSH AF
46E1 2A 57 5A E5        1319          HL=(表ﾎﾟｲﾝﾀ) PUSH HL
46E5                    1320 :
46E5 C5                 1321          PUSH BC
46E6 CD 38 58           1322          [ｶｯｺ開き]
46E9 C1                 1323          POP  BC
46EA                    1324 :
46EA 79                 1325          A=C
46EB FE 0B 28 04 FE 0C 20 1326        IF A=M関$ OR A=未M関$ THEN
46F2 31
46F3 2A 55 5A           1327            HL=(補助DATA)
46F6 7D                 1328            A=L
46F7 FE 01 20 05 CD 52 48 1329          IF A=1 : [引数1]処理
46FE 18 22 FE 02 20 05 CD 1330          EF A=2 : [引数2]処理
4705 5E 48
4707 18 19 FE 03 20 05 CD 1331          EF A=3 : [引数3]処理
470E 6D 48
4710 18 10 FE 04 20 05 CD 1332          EF A=4 : [引数4]処理
4717 C1 48
4719 18 07 FE 05 20 03 CD 1333          EF A=5 : [引数5]処理
4720 D1 48
4722                    1334 :          ELSE  : NOP
4722                    1335            FI
4722                    1336 :
4722 18 2E              1337          ELSE
4724                    1338 :          C:type
4724 AF 32 15 48        1339            (実引数の数)=0
4728 2A 55 5A           1340            HL=(補助DATA)
472B                    1341 :          HL:関数WK
472B E5                 1342            PUSH HL
472C C5 E5              1343              PUSH BC/HL
472E CD AC 5B           1344              [SPCUT]
4731 E1 C1 FE 29 C4 70 47 1345            POP  HL/BC  IF A<>")" [実引数]
4738 E1                 1346            POP  HL
4739                    1347 :          HL:関数WK
4739 23 23 CD 94 1F     1348            INC HL/HL #PEEK     :引数の数
473E                    1349 :
473E FE FF 20 08        1350            IF A=NULL THEN
4742 3A 15 48 CD 9A 1F  1351              A=(実引数の数)  #POKE
4748 18 08              1352            ELSE
474A 47 3A 15 48 B8 C4 27 1353            B=A IF (実引数の数)<>B [引数数ｴﾗｰ]
4751 62
4752                    1354            FI
4752                    1355          FI
4752                    1356 :
4752 CD 36 58           1357          [ｶｯｺ閉じ]
4755                    1358 :
4755 3E 00 32 A4 4F     1359          (初ｺｰﾄﾞ)=FALSE
475A E1 22 57 5A CD 18 48 1360        POP HL (表ﾎﾟｲﾝﾀ)=HL  @[IY]call関数
4761 F1 32 6F 50        1361          POP AF (型)=A
4765                    1362 :        (型)=INT$ or REAL$
4765 3A 6F 50 FE 04 CC DD 1363        IF (型)=REAL$ [TEMP代入処理]
476C 4F
476D                    1364 :
476D C3 BD 50           1365          [PUSH_中間%] RET
4770                    1366 :
4770                    1367 :
4770                    1368 [実引数]  ;in C:type,HL:関数WK
4770 AF 32 16 48        1369          (実引数ﾊﾞｲﾄCNT)=0
4774                    1370 :        HL:関数WKﾎﾟｲﾝﾀ
4774                    1371          {
4774 C5 E5              1372            PUSH BC/HL
4776                    1373 :
4776 3A 16 48 47        1374            B=(実引数ﾊﾞｲﾄCNT)
477A 3A 15 48           1375            A=(実引数の数)
477D 3C FE 09 D4 CF 61  1376            INC A  IF A>引数最大数 [引数過多!]
4783 F5 C5 E5           1377            PUSH AF/BC/HL
4786                    1378 :
4786 CD 6D 3A CD D6 4B  1379              [初論理項] [ｺｰﾄﾞ生成1]
478C CD A2 50           1380              [TOP_ﾏｰｸ]
478F                    1381 :            (型)=INT$ or REAL$
478F E1 C1 F1           1382            POP  HL/BC/AF
4792 32 15 48           1383            (実引数の数)=A
4795 78 32 16 48        1384            (実引数ﾊﾞｲﾄCNT)=B
4799                    1385 :
4799 0C 0D 28 05 79 FE 0A 1386          IF C=0 OR C=未関数$ THEN
47A0 20 16
47A2 FD E5 C1 03        1387              BC=IY  INC BC
47A6                    1388 :
47A6 CD 94 1F 5F 79 CD 5B 1389            #PEEK E=A  A=C #POKE+
47AD 45
47AE CD 94 1F 57 78 CD 5B 1390            #PEEK D=A  A=B #POKE+
47B5 45
47B6                    1391 :
47B6 18 0F              1392            ELSE
47B8 CD 60 45 5F        1393              #PEEK+  E=A
47BC CD 60 45 57        1394              #PEEK+  D=A
47C0                    1395 :            B:(実引数ﾊﾞｲﾄCNT)
47C0 78 80 5F           1396              ADD E,B
47C3 7A CE 00 57        1397              ADC D,0
47C7                    1398            FI
47C7 E5                 1399            PUSH HL
47C8 EB 3E 02 CD 9F 50  1400              EX DE,HL  A=変数% [PUSH_A%HL]
47CE CD E1 50           1401              [EX_STACK]
47D1 CD 31 40           1402              [代入処理]
47D4 CD B4 50           1403              [POP_1]
47D7                    1404 :            (型)=INT$ or REAL$
47D7 E1                 1405            POP  HL
47D8 3A 15 48 5F 16 00 19 1406          E=(実引数の数) D=0  ADD HL,DE  #PEEK
47DF CD 94 1F
47E2 47                 1407            B=A
47E3 FE FF 20 08 3A 6F 50 1408          IF A=NULL  : A=(型) #POKE
47EA CD 9A 1F
47ED 18 09 3A 6F 50 B8 28 1409          EF (型)<>B : [引数型ｴﾗｰ]
47F4 03 CD 35 62
47F8                    1410            FI
47F8                    1411 :
47F8 06 02 3A 6F 50 FE 04 1412          B=2 IF (型)=REAL$ THEN B=(精度)
47FF 20 04 3A AE 69 47
4805 3A 16 48 80 32 16 48 1413          ADD (実引数ﾊﾞｲﾄCNT),B
480C                    1414 :
480C CD B7 37           1415            [ｶﾝﾏ]
480F                    1416 :
480F E1 C1              1417            POP  HL/BC
4811                    1418 :
4811 DA 74 47           1419          } UNTIL NC
4814 C9                 1420          RET
4815                    1421 :
4815 00                 1422 実引数の数    DB 0
4816 00 00              1423 実引数ﾊﾞｲﾄCNT DW $0000
4818                    1424 :
4818                    1425 :
4818                    1426 @[IY]call関数
4818 3E CD CD 7B 5B     1427          A=$CD  @[IY]A
481D                    1428 :        @[IY]関数
481D                    1429 :        RET
481D                    1430 :
481D                    1431 @[IY]関数
481D 2A 57 5A           1432          HL=(表ﾎﾟｲﾝﾀ)
4820 2B                 1433          DEC HL              :補助DATA
4821 2B                 1434          DEC HL              :補助DATA
4822 2B CD 94 1F 57     1435          DEC HL  #PEEK  D=A :表DATA
4827 2B CD 94 1F 5F     1436          DEC HL  #PEEK  E=A :表DATA
482C 2B                 1437          DEC HL              :型
482D 2B CD 94 1F        1438          DEC HL  #PEEK       :TYPE
4831 D5                 1439          PUSH DE
4832                    1440 :
4832 FE 0A 28 04 FE 0C 20 1441        IF A=未関数$ OR A=未M関$ THEN
4839 14
483A FD E5 D1           1442            DE=IY
483D 23                 1443            INC HL
483E 23                 1444            INC HL
483F 7B CD 5B 45        1445            A=E  #POKE+
4843 7A CD 9A 1F        1446            A=D  #POKE
4847 3A E2 39 3C 32 E2 39 1447          INC (未ｺｰﾙ回数)
484E                    1448          FI
484E                    1449 :
484E E1 C3 76 5B        1450          POP HL  @[IY]HL  RET
4852                    1451 :
4852                    1452 :
4852                    1453 [引数1]処理
4852 CD 7C 50           1454          [初ｺｰﾄﾞ初期化]
4855                    1455 :        [引数1]処理SUB
4855                    1456 :        RET
4855                    1457 [引数1]処理SUB
4855 CD 70 3A           1458          [論理項]
4858                    1459 :        [ｺｰﾄﾞ生成1_POP]
4858                    1460 :        RET
4858                    1461 [ｺｰﾄﾞ生成1_POP]
4858 CD D6 4B C3 B4 50  1462          [ｺｰﾄﾞ生成1] [POP_1] RET
485E                    1463 :
485E                    1464 [引数2]処理
485E CD 6D 3A CD F3 48 CD 1465        [初論理項] [ｶﾝﾏ!] [論理項]
4865 70 3A
4867                    1466 :        [TEMPなしｺｰﾄﾞ生成2_POP]
4867                    1467 :        RET
4867                    1468 [TEMPなしｺｰﾄﾞ生成2_POP]
4867 CD 45 4C C3 B4 50  1469          [TEMPなしｺｰﾄﾞ生成2] [POP_1] RET
486D                    1470 :
486D                    1471 [引数3]処理
486D CD 6D 3A CD F3 48 CD 1472        [初論理項] [ｶﾝﾏ!] [TOP_ﾏｰｸ]
4874 A2 50
4876 FE 00 28 32 FE 03 28 1473        IF A<>中間% AND A<>ｺｰﾄﾞ% THEN
487D 2E
487E                    1474 :
487E CD 70 3A CD F3 48 CD 1475          [論理項] [ｶﾝﾏ!] [TOP_ﾏｰｸ]
4885 A2 50
4887 FE 00 28 13 FE 03 28 1476          IF A<>中間% AND A<>ｺｰﾄﾞ% THEN
488E 0F
488F CD 52 48 CD 44 5B 02 1477            [引数1]処理 @[IY]SP DB 2  BC=HL
4896 44 4D
4898 CD 7C 50           1478              [初ｺｰﾄﾞ初期化]
489B C3 67 48           1479              [TEMPなしｺｰﾄﾞ生成2_POP] RET
489E                    1480            FI
489E                    1481 :
489E CD 67 48           1482            [TEMPなしｺｰﾄﾞ生成2_POP]
48A1 CD 44 5B 02 E5 D5  1483            @[IY]SP DB 2  PUSH HL/DE
48A7 CD 52 48           1484            [引数1]処理
48AA                    1485 :
48AA 18 0C              1486          ELSE
48AC CD B4 50           1487            [POP_1]
48AF CD 55 48 CD F3 48  1488            [引数1]処理SUB [ｶﾝﾏ!]
48B5 CD 55 48           1489            [引数1]処理SUB
48B8                    1490          FI
48B8 CD 44 5B 04 44 4D D1 1491        @[IY]SP DB 4  BC=HL  POP DE/HL
48BF E1
48C0 C9                 1492          RET
48C1                    1493 :
48C1                    1494 [引数4]処理
48C1 06 03 CD E3 48     1495          B=3 [引数n]処理
48C6 CD 44 5B           1496          @[IY]SP
48C9 06 E5 DD E1 C1 D1 E1 1497        DB 6  PUSH HL  POP IX/BC/DE/HL
48D0 C9                 1498          RET
48D1                    1499 :
48D1                    1500 [引数5]処理
48D1 06 04 CD E3 48     1501          B=4 [引数n]処理
48D6 CD 44 5B           1502          @[IY]SP
48D9 08 E5 FD E1 DD E1 C1 1503        DB 8  PUSH HL  POP IY/IX/BC/DE/HL
48E0 D1 E1
48E2 C9                 1504          RET
48E3                    1505
48E3                    1506 [引数n]処理
48E3 C5                 1507          PUSH BC
48E4 CD 52 48           1508          [引数1]処理
48E7 C1                 1509          POP  BC
48E8                    1510          DO B {
48E8 C5                 1511            PUSH BC
48E9 CD F3 48 CD 55 48  1512            [ｶﾝﾏ!]  [引数1]処理SUB
48EF C1                 1513            POP  BC
48F0 10 F6              1514          }
48F2 C9                 1515          RET
48F3                    1516 :
48F3                    1517 :
48F3                    1518 [ｶﾝﾏ!]
48F3 3E 2C C3 03 60     1519          A="," [INCIX] RET
48F8                    1520 :
48F8                    1521 ;************************************
48F8                    1522 :
48F8                    1523 [加算処理]
48F8 21 20 FF           1524          HL=#ADD+$FF00
48FB 11 9E DF           1525          DE=[加算]-&R
48FE 01 44 49 C3 BF 4A  1526          BC=[加算最適化]  [二項演算最適化] RET
4904                    1527 :
4904                    1528 [減算処理]
4904 21 23 FF           1529          HL=#SUB+$FF00
4907 11 A0 DF           1530          DE=[減算]-&R
490A 01 2E 49 C3 BF 4A  1531          BC=[減算最適化]  [二項演算最適化] RET
4910                    1532 :
4910                    1533 [乗算処理]
4910 21 26 FF           1534          HL=#MUL+$FF00
4913 11 99 00           1535          DE=&[乗算]-&R
4916 01 A0 49 C3 BF 4A  1536          BC=[乗算最適化]  [二項演算最適化] RET
491C                    1537 :
491C                    1538 [除算処理]
491C 21 29 FF           1539          HL=#DIV+$FF00
491F 11 C3 00           1540          DE=&[除算]-&R
4922 01 6C 4A C3 BF 4A  1541          BC=[除算最適化]  [二項演算最適化] RET
4928                    1542 :
4928                    1543 :
4928 19 C9              1544 [加算] ADD HL,DE  RET
492A B7 ED 52 C9        1545 [減算] SUB HL,DE  RET
492E                    1546 :
492E                    1547 :
492E                    1548 [減算最適化]
492E CD A2 50           1549          [TOP_ﾏｰｸ]
4931 FE 01 28 06        1550          IF A<>定数% THEN
4935 CD 26 4C C3 63 58  1551            [ｺｰﾄﾞ生成2]  @[IY]SUB  RET
493B                    1552          FI
493B CD D0 50 CD A7 6A CD 1553        [POP_DATA] &[単項-] [PUSH_DATA]
4942 C4 50
4944                    1554 :        [加算最適化]
4944                    1555 :        RET
4944                    1556 :
4944                    1557 [加算最適化]
4944 CD 5E 4C           1558          [INT逆順最適化]
4947                    1559 :
4947 CD A2 50 FE 01 C2 8C 1560        [TOP_ﾏｰｸ]  IF A<>定数% [加算ｺｰﾄﾞ] RET
494E 49
494F                    1561 :定数%
494F CD B4 50           1562          [POP_1]
4952                    1563 :0
4952 7C B5 C8           1564          IF HL=0 RET
4955                    1565 :1,2,3   -> INC HL
4955 45 0E 23 7D D6 04 7C 1566        B=L  C=$23 IF HL<4 [加算最適化SUB] RET
495C DE 00 DA 94 49
4961                    1567 :-1,-2,-3 -> DEC HL
4961 7D ED 44 47 0E 2B  1568          A=L NEG B=A  C=$2B
4967 7D D6 FD 7C DE FF D2 1569        IF HL>=$FFFD [加算最適化SUB] RET
496E 94 49
4970                    1570 :
4970 2C 2D 20 15        1571          IF L=0 THEN
4974                    1572 :$100,$200,$300,$400
4974 44 0E 24 7C FE 05 DA 1573          B=H C=$24 IF H<5 [加算最適化SUB] RET
497B 94 49
497D                    1574 :-$100,-$200,-$300,-$400
497D 7C ED 44 47 0E 25  1575            A=H NEG B=A  C=$25
4983 7C FE FC D2 94 49  1576            IF H>=$FC [加算最適化SUB] RET
4989                    1577          FI
4989                    1578 :
4989 CD CE 4B           1579          [PUSH_定数%HL]
498C                    1580 :        [加算ｺｰﾄﾞ]
498C                    1581 :        RET
498C                    1582 :
498C                    1583 [加算ｺｰﾄﾞ]
498C CD 34 4C CD 3F 5B 19 1584        [ｺｰﾄﾞ生成3] @[IY]SP1 ADD HL,DE
4993 C9                 1585          RET
4994                    1586 :
4994                    1587 [加算最適化SUB]
4994 C5                 1588          PUSH BC
4995 CD D6 4B           1589          [ｺｰﾄﾞ生成1]
4998 C1                 1590          POP  BC
4999 79 CD 7B 5B 10 FA  1591          DO B { A=C  @[IY]A }
499F C9                 1592          RET
49A0                    1593 :
49A0                    1594 :
49A0                    1595 [乗算最適化]
49A0 CD 5E 4C           1596          [INT逆順最適化]
49A3                    1597 :
49A3 CD A2 50           1598          [TOP_ﾏｰｸ]
49A6 FE 01 20 45        1599          IF A=定数% THEN
49AA CD B4 50           1600            [POP_1]
49AD                    1601 :0
49AD 7C B5 20 18        1602            IF HL=0 THEN
49B1 CD A2 50           1603              [TOP_ﾏｰｸ]
49B4 FE 00 20 08        1604              IF A=中間% THEN
49B8 CD 44 5B 03 21 00 00 1605              @[IY]SP DB 3 HL=0
49BF C9                 1606                RET
49C0                    1607              FI
49C0 CD B4 50           1608              [POP_1]
49C3 21 00 00 C3 CE 4B  1609              HL=0 [PUSH_定数%HL] RET
49C9                    1610            FI
49C9                    1611 :1
49C9 7C FE 00 20 03 7D FE 1612          IF HL=1 RET
49D0 01 C8
49D2                    1613 :2～16
49D2 7D D6 11 7C DE 00 30 1614          IF HL<=16 THEN
49D9 0C
49DA E5 CD D6 4B E1     1615              PUSH HL  [ｺｰﾄﾞ生成1]  POP HL
49DF                    1616 :
49DF AF 32 28 4A C3 F8 49 1617            (奇数ﾌﾗｸﾞ)=0 [乗算展開] RET
49E6                    1618            FI
49E6                    1619 :2^n
49E6 CD 4E 4A DA 29 4A  1620            [2^n?] IF C [乗算2^n] RET
49EC                    1621 :
49EC CD CE 4B           1622            [PUSH_定数%HL]
49EF                    1623          FI
49EF                    1624 :その他
49EF CD 34 4C           1625          [ｺｰﾄﾞ生成3]
49F2 21 99 00 C3 73 4F  1626          HL=&[乗算]-&R  @[IY]call_RT  RET
49F8                    1627 :
49F8                    1628 [乗算展開]
49F8 CB 45              1629          BIT 0,L
49FA                    1630 :奇数
49FA 28 18              1631          IF NZ THEN
49FC E5                 1632            PUSH HL
49FD 21 28 4A           1633            HL=奇数ﾌﾗｸﾞ
4A00 34 35 20 07        1634            IF (HL)=0 THEN
4A04 35                 1635              DEC (HL)
4A05 CD 44 5B 02 54 5D  1636              @[IY]SP DB 2 DE=HL
4A0B                    1637            FI
4A0B E1 2B CD F8 49     1638            POP  HL  DEC HL [乗算展開]
4A10 3E 19              1639            A=$19
4A12                    1640 :偶数
4A12 18 11              1641          ELSE
4A14 CB 3C CB 1D 7C FE 00 1642          SRL H  RR L  IF HL<>1 [乗算展開]
4A1B 20 03 7D FE 01 C4 F8
4A22 49
4A23 3E 29              1643            A=$29
4A25                    1644          FI
4A25 C3 7B 5B           1645          @[IY]A RET
4A28                    1646 :
4A28 00                 1647 奇数ﾌﾗｸﾞ  DB 0
4A29                    1648 :
4A29                    1649 [乗算2^n]
4A29 C5 CD D6 4B C1     1650          PUSH BC  [ｺｰﾄﾞ生成1]  POP BC
4A2E                    1651 :
4A2E 78 FE 08 38 0D     1652          IF B>=8 THEN
4A33 C5                 1653            PUSH BC
4A34 CD 44 5B 03 65 2E 00 1654            @[IY]SP DB 3 H=L L=0
4A3B C1                 1655            POP  BC
4A3C 78 D6 08 47        1656            SUB B,8
4A40                    1657          FI
4A40                    1658 :
4A40 04 05 28 09        1659          WHILE B<>0 {
4A44 C5                 1660            PUSH BC
4A45 CD 3F 5B 29        1661              @[IY]SP1 : ADD HL,HL
4A49 C1                 1662            POP  BC
4A4A 05                 1663            DEC B
4A4B 18 F3              1664          }
4A4D C9                 1665          RET
4A4E                    1666 :
4A4E                    1667 [2^n?]
4A4E 54 5D              1668          DE=HL
4A50 0E 00              1669          C=0
4A52 06 10              1670          DO B,16 {
4A54 7A FE 00 20 03 7B FE 1671          IF DE=1 THEN B=C SCF RET
4A5B 01 20 03 41 37 C9
4A61 CB 3A              1672            SRL D
4A63 CB 1B 38 03        1673            RR  E  IF C EXIT
4A67 0C                 1674            INC C
4A68 10 EA              1675          }
4A6A B7                 1676          RCF
4A6B C9                 1677          RET
4A6C                    1678 :
4A6C                    1679 :
4A6C                    1680 [除算最適化]
4A6C CD A2 50           1681          [TOP_ﾏｰｸ]
4A6F FE 01 20 1A        1682          IF A=定数% THEN
4A73 CD B4 50           1683            [POP_1]
4A76                    1684 :0
4A76 7C B5 CA 7F 62     1685            IF HL=0 [0除算] RET
4A7B                    1686 :1
4A7B 7C FE 00 20 03 7D FE 1687          IF HL=1 RET
4A82 01 C8
4A84                    1688 :2^n
4A84 CD 4E 4A DA 96 4A  1689            [2^n?] IF C [除算2^n] RET
4A8A                    1690 :
4A8A CD CE 4B           1691            [PUSH_定数%HL]
4A8D                    1692          FI
4A8D                    1693 :その他
4A8D CD 26 4C           1694          [ｺｰﾄﾞ生成2]
4A90 21 C3 00 C3 73 4F  1695          HL=&[除算]-&R  @[IY]call_RT  RET
4A96                    1696 :
4A96                    1697 [除算2^n]
4A96 C5 CD 26 4C C1     1698          PUSH BC [ｺｰﾄﾞ生成2] POP BC
4A9B                    1699 :
4A9B 78 FE 08 38 0D     1700          IF B>=8 THEN
4AA0 C5                 1701            PUSH BC
4AA1 CD 44 5B 03 6C 26 00 1702            @[IY]SP DB 3 L=H H=0
4AA8 C1                 1703            POP  BC
4AA9 78 D6 08 47        1704            SUB B,8
4AAD                    1705          FI
4AAD                    1706 :
4AAD 04 05 28 0D        1707          WHILE B<>0 {
4AB1 C5                 1708            PUSH BC
4AB2 CD 44 5B 04 CB 3C CB 1709            @[IY]SP DB 4 SRL H RR L
4AB9 1D
4ABA C1                 1710            POP  BC
4ABB 05                 1711            DEC B
4ABC 18 EF              1712          }
```

▶大戦略III'90がこんなに早く出るとは。でも今年中に出るのかな。
内藤　剛克(21)愛知県

```
4ABE C9                 1713          RET
4ABF                    1714 :
4ABF                    1715 :
4ABF                    1716 [二項演算最適化]
4ABF                    1717 :        BC:最適化処理ｱﾄﾞﾚｽ
4ABF                    1718 :        HL:実数演算処理ｱﾄﾞﾚｽ-&R
4ABF                    1719 :        DE:整数演算処理ｱﾄﾞﾚｽ-&R
4ABF CD 7B 4C D2 2C 4B  1720          [INTINTﾁｪｯｸ]  IF NC [二項演算2] RET
4AC5 CD 61 4B DA 2C 4B  1721          [定数定数ﾁｪｯｸ] IF C  [二項演算2] RET
4ACB                    1722 :
4ACB C3 8F 4F           1723          CALL [BC] RET
4ACE                    1724 :
4ACE                    1725 :****************************************
4ACE                    1726 :
4ACE                    1727 [単項演算1]
4ACE                    1728 :        HL:実数演算処理ｱﾄﾞﾚｽ-&R
4ACE                    1729 :        DE:整数演算処理ｱﾄﾞﾚｽ-&R
4ACE E5 D5              1730          PUSH HL/DE
4AD0 CD A2 50           1731          [TOP_ﾏｰｸ]
4AD3 D1 E1              1732          POP  DE/HL
4AD5 FE 01 CA 7C 4B     1733          IF A=定数% [単項演算定数] RET
4ADA                    1734 :
4ADA E5 D5              1735          PUSH HL/DE
4ADC                    1736 :
4ADC F5                 1737          PUSH AF
4ADD CD D6 4B           1738          [ｺｰﾄﾞ生成1]
4AE0 F1                 1739          POP  AF
4AE1 FE 00 28 0A 3A 6F 50 1740        IF A<>中間% AND (型)=REAL$ THEN
4AE8 FE 04 20 03
4AEC CD DD 4F           1741            [TEMP代入処理]
4AEF                    1742          FI
4AEF                    1743 :
4AEF 18 49              1744          JR [二演2処理]1
4AF1                    1745 :
4AF1                    1746 :
4AF1                    1747 [単項演算1INT] :結果がINTになる単項演算
4AF1                    1748 :        HL:実数演算処理ｱﾄﾞﾚｽ-&R
4AF1                    1749 :        DE:整数演算処理ｱﾄﾞﾚｽ-&R
4AF1 E5 D5              1750          PUSH HL/DE
4AF3 CD A2 50           1751          [TOP_ﾏｰｸ]
4AF6 D1 E1              1752          POP  DE/HL
4AF8 FE 01 CA 7C 4B     1753          IF A=定数% [単項演算定数] RET
4AFD                    1754 :
4AFD E5 D5              1755          PUSH HL/DE
4AFF                    1756 :
4AFF CD 58 48           1757          [ｺｰﾄﾞ生成1_POP]
4B02                    1758 :
4B02 3A 6F 50           1759          A=(型)
4B05 F5                 1760          PUSH AF
4B06 3E 03 32 6F 50     1761          (型)=INT$
4B0B F1 18 32           1762          POP  AF  JR [二演2処理]2
4B0E                    1763 :
4B0E                    1764 :
4B0E                    1765 [二項演算2INT] :結果がINTになる2項演算
4B0E 01 45 4C           1766          BC=[TEMPなしｺｰﾄﾞ生成2]
4B11 CD 2F 4B           1767          [二演2処理]
4B14                    1768 :
4B14 CD 54 50           1769          [POP_1]
4B17 F5                 1770          PUSH AF
4B18 3E 03 32 6F 50     1771          (型)=INT$
4B1D F1 FE 03 20 02 3E 00 1772        POP AF  IF A=ｺｰﾄﾞ% THEN A=中間%
4B24                    1773 :
4B24 C3 BF 50           1774          [PUSH_A%HL] RET
4B27                    1775 :
4B27                    1776 :
4B27                    1777 [二項演算3]
4B27 01 34 4C 18 03     1778          BC=[ｺｰﾄﾞ生成3] JR [二演2処理]
4B2C                    1779 :
4B2C                    1780 [二項演算2]
4B2C 01 26 4C           1781          BC=[ｺｰﾄﾞ生成2]
4B2F                    1782 :
4B2F                    1783 [二演2処理]
4B2F                    1784 :        BC:[ｺｰﾄﾞ生成n]
4B2F                    1785 :        HL:実数演算処理ｱﾄﾞﾚｽ-&R
4B2F                    1786 :        DE:整数演算処理ｱﾄﾞﾚｽ-&R
4B2F                    1787 :        PUSH BC
4B2F CD 61 4B           1788          [定数定数ﾁｪｯｸ]
4B32                    1789 :        POP  BC
4B32 DA 86 4B           1790          IF C [二項演算定定] RET
4B35                    1791 :
4B35 E5 D5              1792          PUSH HL/DE
4B37                    1793 :
4B37 CD 8F 4F           1794          CALL [BC] :[ｺｰﾄﾞ生成n]
4B3A                    1795 :
4B3A                    1796 [二演2処理]1
4B3A CD B4 50           1797          [POP_1]
4B3D 3A 6F 50           1798          A=(型)
4B40                    1799 [二演2処理]2
4B40 FE 03 20 04        1800          IF A=INT$ THEN
4B44 E1 F1              1801            POP HL/AF
4B46 18 07              1802          ELSE
4B48 F1 E1              1803            POP AF/HL
4B4A 7C B5 CC BA 61     1804            IF HL=0 [REAL不可!]
4B4F                    1805          FI
4B4F 7C FE FF 20 07 26 00 1806        IF H=$FF : H=0  @[IY]call_SRB
4B56 CD 7D 4F
4B59 18 03 CD 73 4F     1807          ELSE     :      @[IY]call_RT
4B5E                    1808          FI
4B5E                    1809 :        (型)=INT$ or REAL$
4B5E C3 BD 50           1810          [PUSH_中間%]  RET
4B61                    1811 :
4B61                    1812 :
4B61                    1813 [定数定数ﾁｪｯｸ]
4B61 E5 D5 C5           1814          PUSH HL/DE/BC
4B64 CD A2 50 5F        1815          [TOP_ﾏｰｸ] E=A
4B68 CD A7 50           1816          [SEC_ﾏｰｸ]
4B6B FE 01 20 08 7B FE 01 1817        IF A=定数% AND E=定数% : SCF
4B72 20 03 37
4B75 18 01 B7           1818          ELSE                   : RCF
4B78                    1819          FI
4B78 C1 D1 E1           1820          POP  BC/DE/HL
4B7B C9                 1821          RET
4B7C                    1822 :
4B7C                    1823 :
4B7C                    1824 [単項演算定数]
4B7C E5 D5              1825          PUSH HL/DE
4B7E                    1826 :
4B7E CD B4 50 11 00 00  1827          [POP_1] DE=0
4B84                    1828 :        HL:第1項
4B84 18 0D              1829          JR [二項演算定定]1
4B86                    1830 :
4B86                    1831 :
4B86                    1832 [二項演算定定]
4B86 E5 D5              1833          PUSH HL/DE
4B88 CD 9C 4C           1834          [型変換]
4B8B                    1835 :
4B8B CD B4 50 E5        1836          [POP_1] PUSH HL
4B8F CD B4 50 D1        1837          [POP_1] POP  DE
4B93                    1838 :        HL:第1項  DE:第2項
4B93                    1839 [二項演算定定]1
4B93 3A 6F 50 FE 03 20 04 1840        IF (型)=INT$ THEN
4B9A C1 F1              1841            POP BC/AF
4B9C 18 07              1842          ELSE
4B9E F1 C1              1843            POP AF/BC
4BA0 78 B1 CC BA 61     1844            IF BC=0 [REAL不可!]
4BA5                    1845          FI
4BA5                    1846 :        HL:第1項  DE:第2項
4BA5 E5                 1847          PUSH HL
4BA6 78 FE FF 20 07 06 00 1848        IF B=$FF : B=0  HL=(SOROBANｱﾄﾞﾚｽ)
4BAD 2A CC 33
4BB0 18 03 21 8A 69     1849          ELSE     :      HL=&RUNTIME
4BB5                    1850          FI
4BB5 79 85 4F 78 8C 47  1851          ADD BC,HL
4BBB E1 CD 8F 4F        1852          POP HL    CALL [BC]
4BBF                    1853 :
4BBF 3A 6F 50 FE 04 20 08 1854        IF (型)=REAL$ AND DE<>0 THEN
4BC6 7A B3 28 04
4BCA                    1855 :          DE:第2項のｱﾄﾞﾚｽ
4BCA ED 53 5F 44        1856            (定数ﾎﾟｲﾝﾀ)=DE     :定数ｽﾀｯｸの解放
4BCE                    1857          FI
4BCE                    1858 :        (型)=INT$ or REAL$
4BCE                    1859 :        [PUSH_定数%HL]
4BCE                    1860 :        RET
4BCE                    1861 :
4BCE                    1862 [PUSH_定数%HL]
4BCE 3E 01 C3 BF 50     1863          A=定数%  [PUSH_A%HL] RET
4BD3                    1864 :
4BD3                    1865 :
4BD3                    1866 [整数ｺｰﾄﾞ生成1]
4BD3 CD 2E 4D           1867          [TOP整数化]
4BD6                    1868 :        [ｺｰﾄﾞ生成1]
4BD6                    1869 :        RET
4BD6                    1870 :
4BD6                    1871 [ｺｰﾄﾞ生成1]
4BD6 CD 9B 50           1872          [POP_ﾏｰｸ]
4BD9 F5                 1873          PUSH AF
4BDA                    1874 :
4BDA 08                 1875          EX AF,AF'
4BDB                    1876 :
4BDB 3A 6F 50 FE 03 20 1E 1877        IF (型)=INT$ THEN
4BE2 08                 1878            EX AF,AF'
4BE3 FE 01 20 08 CD 91 4F 1879          IF A=定数% : @PUSH_HL @HLdata
4BEA CD 6A 5B
4BED 18 0F FE 02 20 08 CD 1880          EF A=変数% : @PUSH_HL @HL[data]
4BF4 91 4F CD 63 5B
4BF9 18 03 CD D0 50     1881            ELSE       : [POP_DATA]
4BFE                    1882            FI
4BFE                    1883 :
4BFE 18 1C              1884          ELSE
4C00 08                 1885            EX AF,AF'
4C01 FE 01 20 08 CD 91 4F 1886          IF A=定数% : @PUSH_HL @HL確定
4C08 CD A1 44
4C0B 18 0F FE 02 20 08 CD 1887          EF A=変数% : @PUSH_HL @HLdata
4C12 91 4F CD 6A 5B
4C17 18 03 CD D0 50     1888            ELSE       : [POP_DATA]
4C1C                    1889            FI
4C1C                    1890          FI
4C1C                    1891 :
4C1C F1 FE 03 28 02 3E 00 1892        POP AF  IF A<>ｺｰﾄﾞ% THEN A=中間%
4C23                    1893 :
4C23 C3 BF 50           1894          [PUSH_A%HL] RET
4C26                    1895 :
4C26                    1896 :
4C26                    1897 [ｺｰﾄﾞ生成2]
4C26 3E 01 32 56 50     1898          (TEMPﾌﾗｸﾞ)=TRUE
4C2B                    1899 :
4C2B CD 9C 4C           1900          [型変換]
4C2E                    1901 :
4C2E CD DC 4D C3 BD 50  1902          [二項TBLｺｰﾙ]  [PUSH_中間%] RET
4C34                    1903 :
4C34                    1904 :
4C34                    1905 [ｺｰﾄﾞ生成3]
4C34 3E 01 32 56 50     1906          (TEMPﾌﾗｸﾞ)=TRUE
4C39                    1907 :
4C39 CD 9C 4C           1908          [型変換]
4C3C                    1909 :
4C3C CD 5E 4C           1910          [INT逆順最適化]
4C3F                    1911 :
4C3F CD DC 4D C3 BD 50  1912          [二項TBLｺｰﾙ]  [PUSH_中間%] RET
4C45                    1913 :
4C45                    1914 :
4C45                    1915 [TEMPなしｺｰﾄﾞ生成2]
4C45                    1916 :結果がINTになる2項演算用
4C45 3E 00 32 56 50     1917          (TEMPﾌﾗｸﾞ)=FALSE
4C4A                    1918 :
4C4A CD 9C 4C           1919          [型変換]
4C4D                    1920 :
4C4D CD DC 4D C3 BD 50  1921          [二項TBLｺｰﾙ]  [PUSH_中間%] RET
4C53                    1922 :
4C53                    1923 :
4C53                    1924 [無変換ｺｰﾄﾞ生成2]
4C53                    1925 :型変換は行わない。配列代入用
4C53 3E 00 32 56 50     1926          (TEMPﾌﾗｸﾞ)=FALSE
4C58                    1927 :
4C58 CD DC 4D C3 BD 50  1928          [二項TBLｺｰﾙ]  [PUSH_中間%] RET
4C5E                    1929 :
4C5E                    1930 :
4C5E                    1931 [INT逆順最適化]
4C5E CD 7B 4C D0        1932          [INTINTﾁｪｯｸ] IF NC RET
4C62                    1933 :
4C62 CD A2 50 FE 02 CC E1 1934        [TOP_ﾏｰｸ] IF A=変数% [EX_STACK]
4C69 50
4C6A CD A2 50 FE 00 CC E1 1935        [TOP_ﾏｰｸ] IF A=中間% [EX_STACK]
4C71 50
4C72 CD A7 50 FE 01 CC E1 1936        [SEC_ﾏｰｸ] IF A=定数% [EX_STACK]
4C79 50
4C7A C9                 1937          RET
4C7B                    1938 :
4C7B                    1939 :
4C7B                    1940 [INTINTﾁｪｯｸ]
4C7B C5 D5 E5           1941          PUSH BC/DE/HL
4C7E                    1942 :
4C7E CD A2 50 3A 6F 50 47 1943        [TOP_ﾏｰｸ] B=(型)          ;第2項
4C85 CD A7 50 3A 6F 50  1944          [SEC_ﾏｰｸ] A=(型)          ;第1項
4C8B                    1945 :
4C8B FE 03 20 08 78 FE 03 1946        IF A=INT$ AND B=INT$ : SCF
4C92 20 03 37
4C95 18 01 B7           1947          ELSE                 : RCF
4C98                    1948          FI
4C98                    1949 :
4C98 E1 D1 C1           1950          POP HL/DE/BC
4C9B C9                 1951          RET
4C9C                    1952 :
4C9C                    1953 :
4C9C                    1954 [型変換]
4C9C CD A2 50 5F 3A 6F 50 1955        [TOP_ﾏｰｸ] E=A  C=(型)          :第2項
4CA3 4F
4CA4 CD A7 50 57 3A 6F 50 1956        [SEC_ﾏｰｸ] D=A  B=(型)          :第1項
4CAB 47
4CAC                    1957 :
4CAC                    1958 :->実実
4CAC 78 FE 03 20 0A 79 FE 1959        IF B=INT$  AND C=REAL$ : [整実実数化]
4CB3 04 20 05 CD C9 4C
4CB8 18 0D 78 FE 04 20 08 1960        EF B=REAL$ AND C=INT$  : [TOP実数化]
4CC0 79 FE 03 20 03 CD 08
4CC7 4D
4CC8                    1961          FI
4CC8                    1962 :
4CC8 C9                 1963          RET
4CC9                    1964 :
4CC9                    1965 :
4CC9                    1966 [整実実数化]
4CC9 D5 CD E1 50 D1     1967          PUSH DE  [EX_STACK]  POP DE
4CCE                    1968 :
4CCE 7A FE 01 20 05     1969          IF D=定数% THEN
4CD3 CD 64 4D           1970            [TOP定数実数化]
4CD6                    1971 :
4CD6 18 2D              1972          ELSE
4CD8 7B FE 00 20 17     1973            IF E=中間% THEN
4CDD 7A FE 00 20 06     1974              IF D=中間% THEN
4CE2 CD 3F 5B E3        1975                @[IY]SP1 : EX (SP),HL
4CE5 18 03              1976              ELSE
4CE8 CD D6 4B           1977                [ｺｰﾄﾞ生成1]
4CEB                    1978              FI
4CEB CD E6 4F           1979              [TEMP実数化]
4CEE CD 3F 5B E3        1980              @[IY]SP1   : EX (SP),HL
4CF2                    1981 :
4CF2 18 06              1982            ELSE
4CF4 CD D6 4B           1983              [ｺｰﾄﾞ生成1]
4CF7 CD E6 4F           1984              [TEMP実数化]
4CFA                    1985            FI
4CFA                    1986 :
4CFA CD B4 50 3E 04 32 6F 1987          [POP_1]  (型)=REAL$  [PUSH_中間%]
4D01 50 CD BD 50
4D05                    1988          FI
4D05 C3 E1 50           1989          [EX_STACK] RET
4D08                    1990 :
4D08                    1991 :
4D08                    1992 [ｷｬｽﾄITR]
4D08                    1993 [TOP実数化]
4D08 CD A2 50 5F 3A 6F 50 1994        [TOP_ﾏｰｸ] E=A  IF (型)=REAL$ RET
4D0F FE 04 C8
4D12                    1995 :
4D12 7B FE 01 20 05     1996          IF E=定数% THEN
4D17 CD 64 4D           1997            [TOP定数実数化]
4D1A 18 11              1998          ELSE
4D1C CD D6 4B           1999            [ｺｰﾄﾞ生成1]
4D1F CD E6 4F           2000            [TEMP実数化]
4D22 CD B4 50 3E 04 32 6F 2001          [POP_1]  (型)=REAL$  [PUSH_中間%]
4D29 50 CD BD 50
4D2D                    2002          FI
4D2D C9                 2003          RET
4D2E                    2004 :
4D2E                    2005 :
4D2E                    2006 [ｷｬｽﾄRTI]
4D2E                    2007 [TOP整数化]
4D2E CD A2 50 5F 3A 6F 50 2008        [TOP_ﾏｰｸ] E=A  IF (型)=INT$ RET
4D35 FE 03 C8
4D38                    2009 :
4D38 7B FE 01 20 05     2010          IF E=定数% THEN
4D3D CD A3 4D           2011            [TOP定数整数化]
4D40 18 21              2012          ELSE
4D42 CD D6 4B           2013            [ｺｰﾄﾞ生成1]
4D45 21 1D 00           2014            HL=#CVFTI
4D48 3A 47 3C FE 55 20 03 2015          IF (変換SW)="U" THEN HL=#CVFTU
4D4F 21 1A 00
4D52 CD 7D 4F           2016            @[IY]call_SRB
4D55 CD A5 4F           2017            @[IY]EX
4D58 CD B4 50 3E 03 32 6F 2018          [POP_1]  (型)=INT$  [PUSH_中間%]
4D5F 50 CD BD 50
4D63                    2019          FI
4D63 C9                 2020          RET
4D64                    2021 :
4D64                    2022 :
4D64                    2023 [TOP定数実数化]
4D64 3A 6F 50 F5 C5 D5 E5 2024        A=(型)  PUSH AF/BC/DE/HL
4D6B                    2025 :
4D6B CD A2 50           2026          [TOP_ﾏｰｸ]
4D6E FE 01 20 29 3A 6F 50 2027        IF A=定数% AND (型)=INT$ THEN
4D75 FE 03 20 22
4D79 CD B4 50 EB        2028            [POP_1]  EX DE,HL
4D7D 3A AE 69 CD 36 44  2029            A=(精度) [INC定数ﾎﾟｲﾝﾀ]
4D83 01 14 00           2030            BC=#CVITF
4D86 3A 47 3C FE 55 20 03 2031          IF (変換SW)="U" THEN BC=#CVUTF
4D8D 01 11 00
4D90 CD 87 4F           2032            CALL[SRB]
4D93 3E 04 32 6F 50 CD CE 2033          (型)=REAL$  [PUSH_定数%HL]
4D9A 4B
4D9B                    2034          FI
4D9B                    2035 :
4D9B E1 D1 C1 F1 32 6F 50 2036        POP HL/DE/BC/AF  (型)=A
4DA2 C9                 2037          RET
4DA3                    2038 :
4DA3                    2039 :
4DA3                    2040 [TOP定数整数化]
4DA3 3A 6F 50 F5 C5 D5 E5 2041        A=(型)  PUSH AF/BC/DE/HL
4DAA                    2042 :
4DAA CD A2 50           2043          [TOP_ﾏｰｸ]
4DAD FE 01 20 23 3A 6F 50 2044        IF A=定数% AND (型)=REAL$ THEN
4DB4 FE 04 20 1C
4DB8 CD B4 50           2045            [POP_1]
4DBB 01 1D 00           2046            BC=#CVFTI
4DBE 3A 47 3C FE 55 20 03 2047          IF (変換SW)="U" THEN BC=#CVFTU
4DC5 01 1A 00
4DC8 CD 87 4F           2048            CALL[SRB]
4DCB EB                 2049            EX DE,HL
4DCC 3E 03 32 6F 50 CD CE 2050          (型)=INT$  [PUSH_定数%HL]
4DD3 4B
4DD4                    2051          FI
4DD4                    2052 :
4DD4 E1 D1 C1 F1 32 6F 50 2053        POP HL/DE/BC/AF  (型)=A
4DDB C9                 2054          RET
4DDC                    2055 :
4DDC                    2056 :
4DDC                    2057 [二項TBLｺｰﾙ]
4DDC CD 9B 50 87 5F 3A 6F 2058        [POP_ﾏｰｸ] ADD A,A  E=A D=(型)
4DE3 50 57
4DE5 CD 9B 50 87 4F 3A 6F 2059        [POP_ﾏｰｸ] ADD A,A  C=A B=(型)
4DEC 50 47
4DEE                    2060 :
4DEE 78 FE 04 20 0A 7A FE 2061        IF B=REAL$ AND D=REAL$ : HL=[TBL演算R]
4DF5 04 20 05 21 31 4E
4DFB                    2062 :        EF B=REAL$ AND D=INT$  : HL=[TBL演算RI]
4DFB 18 12 78 FE 03 20 0A 2063        EF B=INT$  AND D=REAL$ : HL=[TBL演算IR]
4E02 7A FE 04 20 05 21 29
4E09 4E
4E0A 18 03 21 21 4E     2064          ELSE                   : HL=[TBL演算I]
4E0F                    2065          FI
4E0F                    2066 :        (型)=INT$ or REAL$
4E0F 06 00 16 00        2067          B=0  D=0
4E13                    2068 :
4E13 19                 2069          ADD HL,DE
4E14 5E 23              2070          E=(HL) INC HL
4E16 56                 2071          D=(HL)
4E17                    2072 :
4E17 EB 09              2073          EX DE,HL  ADD HL,BC
4E19 5E 23              2074          E=(HL) INC HL
4E1B 56 EB              2075          D=(HL) EX DE,HL
4E1D                    2076 :
4E1D CD 81 1F           2077          CALL [HL]
4E20 C9                 2078          RET
4E21                    2079 :
4E21                    2080 :
4E21                    2081 [TBL演算I]
4E21 39 4E 41 4E 49 4E 39 2082              DW [中間I],[定数I],[変数I],[中間I]
4E28 4E
4E29                    2083 [TBL演算IR]
4E29 39 4E 51 4E 41 4E 29 2084              DW [中間I],[定数IR],[定数I],[中間I]
4E30 4E
4E31                    2085 [TBL演算R]
4E31 59 4E 61 4E 69 4E 71 2086              DW [中間R],[定数R],[変数R],[ｺｰﾄﾞR]
4E38 4E
4E39                    2087 :
4E39                    2088 [中間I]
4E39 C7 4E 7F 4E 9A 4E C7 2089              DW [中I中I],[定I中I],[変I中I],[中I中I]
4E40 4E
4E41                    2090 [定数I]
4E41 E0 4E 88 4E A3 4E E0 2091              DW [中I定I],[定I定I],[変I定I],[中I定I]
4E48 4E
4E49                    2092 [変数I]
4E49 79 4E 91 4E AC 4E 79 2093              DW [中I変I],[定I変I],[変I変I],[中I変I]
4E50 4E
4E51                    2094 :
4E51                    2095 [定数IR]
4E51 D7 4E B5 4E BE 4E D7 2096              DW [中I定R],[定I定R],[変I定R],[中I定R]
4E58 4E
4E59                    2097 :
4E59                    2098 [中間R]
4E59 C7 4E E6 4E 25 4F 5D 2099              DW [中R中R],[定R中R],[変R中R],[コR中R]
4E60 4F
4E61                    2100 [定数R]
4E61 D7 4E 07 4F 3F 4F D4 2101              DW [中R定R],[定R定R],[変R定R],[コR定R]
4E68 4E
4E69                    2102 [変数R]
4E69 E0 4E 16 4F 4E 4F DD 2103              DW [中R変R],[定R変R],[変R変R],[コR変R]
4E70 4E
4E71                    2104 [ｺｰﾄﾞR]
4E71 C7 4E E6 4E 25 4F 5D 2105              DW [中RコR],[定RコR],[変RコR],[コRコR]
4E78 4F
4E79                    2106 :
4E79                    2107 :
4E79                    2108 [中I変I]
4E79 CD 51 5B C3 D0 50  2109              @DE[data]  [POP_DATA] RET
4E7F                    2110 :
4E7F                    2111 [定I中I]
4E7F CD D0 50 CD A5 4F C3 2112            [POP_DATA] @[IY]EX    @HLdata   RET
4E86 6A 5B
4E88                    2113 :
4E88                    2114 [定I定I]
4E88 CD 91 4F CD 5C 5B C3 2115            @PUSH_HL  @DEdata     @HLdata   RET
4E8F 6A 5B
4E91                    2116 :
4E91                    2117 [定I変I]
4E91 CD 91 4F CD 51 5B C3 2118            @PUSH_HL  @DE[data]   @HLdata   RET
4E98 6A 5B
4E9A                    2119 :
4E9A                    2120 [変I中I]
4E9A CD D0 50 CD A5 4F C3 2121            [POP_DATA] @[IY]EX    @HL[data] RET
4EA1 63 5B
4EA3                    2122 :
4EA3                    2123 [変I定I]
4EA3 CD 91 4F CD 5C 5B C3 2124            @PUSH_HL  @DEdata     @HL[data] RET
4EAA 63 5B
4EAC                    2125 :
4EAC                    2126 [変I変I]
4EAC CD 91 4F CD 51 5B C3 2127            @PUSH_HL  @DE[data]   @HL[data] RET
4EB3 63 5B
4EB5                    2128 :
4EB5                    2129 :
4EB5                    2130 [定I定R]
4EB5 CD 91 4F CD A7 44 C3 2131            @PUSH_HL  @DE確定     @HLdata   RET
4EBC 6A 5B
4EBE                    2132 :
4EBE                    2133 [変I定R]
4EBE CD 91 4F CD A7 44 C3 2134            @PUSH_HL  @DE確定     @HL[data] RET
4EC5 63 5B
4EC7                    2135 :
4EC7                    2136 :
4EC7                    2137 [中I中I]
4EC7                    2138 [中R中R]
4EC7                    2139 [中RコR]
4EC7 CD A5 4F CD 3F 5B E1 2140            @[IY]EX   @[IY]SP1 : POP HL
4ECE                    2141 :            [POP_DATA2]
4ECE                    2142 :            RET
4ECE                    2143 :
4ECE                    2144 [POP_DATA2]
4ECE CD D0 50 C3 D0 50  2145              [POP_DATA] [POP_DATA] RET
4ED4                    2146 :
4ED4                    2147 [コR定R]
4ED4 CD D7 4F           2148              [TEMP代入]
4ED7                    2149 :            [中R定R]
4ED7                    2150 :            RET
4ED7                    2151 :
4ED7                    2152 [中I定R]
4ED7                    2153 [中R定R]
4ED7 CD A7 44 C3 D0 50  2154              @DE確定    [POP_DATA] RET
4EDD                    2155 :
4EDD                    2156 [コR変R]
4EDD CD D7 4F           2157              [TEMP代入]
4EE0                    2158 :            [中R変R]
4EE0                    2159 :            RET
4EE0                    2160 :
4EE0                    2161 [中R変R]
4EE0                    2162 [中I定I]
4EE0 CD 5C 5B C3 D0 50  2163              @DEdata    [POP_DATA] RET
4EE6                    2164 :
4EE6                    2165 [定R中R]
4EE6                    2166 [定RコR]
4EE6 CD D0 50           2167              [POP_DATA]
4EE9 3A 56 50 FE 01 20 10 2168            IF (TEMPﾌﾗｸﾞ)=TRUE THEN
4EF0 CD 3F 5B E5        2169                @[IY]SP1 : PUSH HL
4EF4 CD A1 44 CD D7 4F  2170                @HL確定  [TEMP代入]
4EFA CD 3F 5B D1        2171                @[IY]SP1 : POP  DE
4EFE                    2172 :
4EFE 18 06              2173              ELSE
4F00 CD A5 4F CD A1 44  2174                @[IY]EX  @HL確定
4F06                    2175              FI
4F06 C9                 2176              RET
4F07                    2177 :
4F07                    2178 [定R定R]
4F07 CD E7 50           2179              [EX_DATA]
4F0A CD 91 4F           2180              @PUSH_HL
4F0D CD A1 44 CD D7 4F C3 2181            @HL確定  [TEMP代入]  @DE確定 RET
4F14 A7 44
4F16                    2182 :
```

▶やっと女ができたのはいいが，ハードディスクを買おうと思って貯金していた７万円を１週間足らずで全部使われてしまった。プログラミングをする時間もない。おそろしや女。

橘　正彦(17)福島県

```
4F16                      2183 [定R変R]
4F16 CD E7 50             2184         [EX_DATA]
4F19 CD 91 4F             2185         @PUSH_HL
4F1C CD A1 44 CD D7 4F C3 2186         @HL確定 [TEMP代入] @DEdata RET
4F23 5C 5B
4F25                      2187 :
4F25                      2188 [変R中R]
4F25                      2189 [変Rコ R]
4F25 3A 56 50 FE 00 CA 7F 2190         IF (TEMPﾌﾗｸﾞ)=FALSE [定I中I] RET
4F2C 4E
4F2D                      2191 :
4F2D CD D0 50             2192         [POP_DATA]
4F30 CD 3F 5B E5          2193         @[IY]SP1 : PUSH HL
4F34 CD 6A 5B CD D7 4F    2194         @HLdata [TEMP代入]
4F3A CD 3F 5B D1          2195         @[IY]SP1 : POP  DE
4F3E C9                   2196         RET
4F3F                      2197 :
4F3F                      2198 [変R定R]
4F3F CD E7 50             2199         [EX_DATA]
4F42 CD 91 4F             2200         @PUSH_HL
4F45 CD 6A 5B CD D7 4F C3 2201         @HLdata [TEMP代入] @DE確定 RET
4F4C A7 44
4F4E                      2202 :
4F4E                      2203 [変R変R]
4F4E CD E7 50             2204         [EX_DATA]
4F51 CD 91 4F             2205         @PUSH_HL
4F54 CD 6A 5B CD D7 4F C3 2206         @HLdata [TEMP代入] @DEdata RET
4F5B 5C 5B
4F5D                      2207 :
4F5D                      2208 [コR中R]
4F5D                      2209 [コRコR]
4F5D 3A 56 50 FE 00 CA C7 2210         IF (TEMPﾌﾗｸﾞ)=FALSE [中R中R] RET
4F64 4E
4F65                      2211 :
4F65 CD 3F 5B E3          2212         @[IY]SP1 : EX (SP).HL
4F69 CD D7 4F             2213         [TEMP代入]
4F6C CD 3F 5B D1          2214         @[IY]SP1 : POP DE
4F70                      2215 :
4F70 C3 CE 4E             2216         [POP_DATA2] RET
4F73                      2217 :
4F73                      2218 :
4F73                      2219 @[IY]call_RT
4F73 D5                   2220         PUSH DE
4F74 ED 5B C3 33 19       2221         DE=(OBJ先頭)     ADD HL.DE
4F79 D1                   2222         POP  DE
4F7A C3 71 5B             2223         @[IY]CALL  RET
4F7D                      2224 :
4F7D                      2225 :
4F7D                      2226 @[IY]call_SRB
4F7D D5                   2227         PUSH DE
4F7E ED 5B CC 33 19       2228         DE=(SOROBANｱﾄﾞﾚｽ) ADD HL.DE
4F83 D1                   2229         POP  DE
4F84 C3 71 5B             2230         @[IY]CALL  RET
4F87                      2231 :
4F87                      2232 :
4F87                      2233 CALL[SRB] ;in BC
4F87 E5                   2234         PUSH HL
4F88 2A CC 33 09 44 4D    2235         HL=(SOROBANｱﾄﾞﾚｽ) ADD HL.BC  BC=HL
4F8E E1                   2236         POP  HL
4F8F                      2237 :       CALL [BC]
4F8F                      2238 :       RET
4F8F                      2239 :
4F8F                      2240 [BC]
4F8F C5 C9                2241         PUSH BC  RET
4F91                      2242 :
4F91                      2243 :
4F91                      2244 @PUSH_HL
4F91 3A A4 4F FE 01 20 07 2245         IF (初ｺｰﾄﾞ)=TRUE THEN
4F98 3E 00 32 A4 4F       2246           (初ｺｰﾄﾞ)=FALSE
4F9D 18 04                2247         ELSE
4F9F CD 3F 5B E5          2248           @[IY]SP1 : PUSH HL
4FA3                      2249         FI
4FA3 C9                   2250         RET
4FA4                      2251 :
4FA4 00                   2252 初ｺｰﾄﾞ  DB 0
4FA5                      2253 :
4FA5                      2254 :
4FA5                      2255 @[IY]EX
4FA5 3E 00                2256         A=$00
4FA6                      2257 最終ｺｰﾄﾞ  EQU $-1
4FA7 FE EB 20 05          2258         IF A=EXS THEN
4FAB FD 2B                2259           DEC IY
4FAD                      2260 [最終ｺｰﾄﾞ0]
4FAD AF                   2261           A=0
4FAE 18 07                2262         ELSE
4FB0 3E EB CD 7B 5B 3E EB 2263           A=$EB @[IY]A  A=EXS
4FB7                      2264         FI
4FB7 32 A6 4F C9          2265         (最終ｺｰﾄﾞ)=A  RET
4FBB                      2266 :
00EB                      2267 EXS EQU $EB
4FBB                      2268 :
4FBB                      2269 :
4FBB                      2270 [TEMP初期化]
4FBB 21 57 50             2271         HL=TEMPﾜｰｸ
4FBE 06 08                2272         DO B.TEMP数 {
4FC0 36 00 23 23 23       2273           (HL)=FALSE  INC HL/HL/HL
4FC5 10 F9                2274         }
4FC7 C9                   2275         RET
4FC8                      2276 :
4FC8                      2277 :
4FC8                      2278 [TEMPﾁｪｲﾝ初期化]
4FC8 21 57 50             2279         HL=TEMPﾜｰｸ
4FCB 06 08                2280         DO B.TEMP数 {
4FCD 23                   2281                 INC HL
4FCE 36 00 23             2282           (HL)=0 INC HL
4FD1 36 00 23             2283           (HL)=0 INC HL
4FD4 10 F7                2284         }
4FD6 C9                   2285         RET
4FD7                      2286 :
4FD7                      2287 :
4FD7                      2288 [TEMP代入]
4FD7 3A 56 50 FE 00 C8    2289         IF (TEMPﾌﾗｸﾞ)=FALSE RET
4FDD                      2290 :       [TEMP代入処理]
4FDD                      2291 :       RET
4FDD                      2292 :
4FDD                      2293 [TEMP代入処理]
4FDD CD F9 4F             2294         @[IY]DEtemp
4FE0 21 8C 00 C3 73 4F    2295         HL=&MOVE_EX-&R  @[IY]call_RT  RET
4FE6                      2296 :
4FE6                      2297 :
4FE6                      2298 [TEMP実数化]
4FE6 CD F9 4F             2299         @[IY]DEtemp
4FE9 21 95 00             2300         HL=&CVITF-&R
4FEC 3A 47 3C FE 55 20 03 2301         IF (変換SW)="U" THEN HL=&CVUTF-&R
4FF3 21 91 00
4FF5 C3 73 4F             2302         @[IY]call_RT  RET
4FF9                      2303 :
4FF9                      2304 :
4FF9                      2305 @[IY]DEtemp
4FF9                      2306 :
4FF9                      2307 :TEMP
4FF9 21 57 50             2308         HL=TEMPﾜｰｸ
4FFC 7E FE 00 28 0E       2309         UNTIL (HL)=FALSE {
5001 23 23 23             2310           INC HL/HL/HL
5004                      2311 :
5004 7D D6 6F 7C DE 50 D4 2312           IF HL>=TEMPﾜｰｸEND [TEMP_OVER!]
500B 91 61
500D 18 ED                2313         }
500F 36 01 23             2314         (HL)=TRUE INC HL
5012                      2315 :
5012 FD E5 D1 13          2316         DE=IY    INC DE
5016 7E 73 5F 23          2317         EX (HL).E INC HL
501A 7E 72 57             2318         EX (HL).D
501D EB                   2319         EX DE.HL
501E                      2320 :
501E                      2321 :
501E C3 5F 5B             2322         @[IY]DEnn RET
5021                      2323 :
5021                      2324 :
5021                      2325 [TEMPﾁｪｲﾝ]
5021 21 57 50             2326         HL=TEMPﾜｰｸ
5024 06 08                2327         DO B.TEMP数 {
5026 C5                   2328           PUSH BC
5027                      2329 :
5027 23                   2330           INC HL
5028 5E 23                2331           E=(HL) INC HL
502A 56 23                2332           D=(HL) INC HL
502C                      2333 :         PUSH HL
502C 7A B3 28 0D          2334           IF DE<>0 THEN
5030 CD 41 50             2335             [ﾁｪｲﾝ処理]
5033                      2336 :
5033 3A AE 69 47 AF CD 7B 2337             DO B.(精度) { A=0  @[IY]A }
503A 5B 10 FA
503D                      2338           FI
503D                      2339 :         POP HL
503D C1                   2340           POP BC
503E 10 E6                2341         }
5040 C9                   2342         RET
5041                      2343 :
5041                      2344 :
5041                      2345 [ﾁｪｲﾝ処理] ;in DE
5041 E5 C5                2346         PUSH HL/BC
5043                      2347 :
5043 EB FD E5 C1          2348         EX DE.HL  BC=IY
5047                      2349 :
```

```
5047 7C B5 28 08          2350         UNTIL HL=0 {
504B                      2351 :
504B CD 81 69             2352           CALL @PATCH1
504E                      2353 :         E=(HL) (HL)=C  INC HL
504E 56 70                2354           D=(HL) (HL)=B
5050 EB                   2355           EX DE.HL
5051 18 F4                2356         }
5053                      2357 :
5053 C1 E1                2358         POP BC/HL
5055 C9                   2359         RET
5056                      2360 :
5056 00                   2361 TEMPﾌﾗｸﾞ   DB 0
5057 00 00 00 00 00 00 00 2362 TEMPﾜｰｸ    DS TEMP数*3
505E 00 00 00 00 00 00 00
5065 00 00 00 00 00 00 00
506C 00 00 00
506F                      2363 TEMPﾜｰｸEND
506F                      2364 :
506F                      2365 ;****************************************
506F                      2366 :
506F 00                   2367 型  DB 0
5070                      2368 :
5070                      2369 [式ｽﾀｯｸ初期化]
5070 21 FE 50 22 FA 50    2370         (ﾏｰｸﾎﾟｲﾝﾀ)=HL=式ｽﾀｯｸM
5076 21 26 51 22 FC 50    2371         (DATAﾎﾟｲﾝﾀ)=HL=式ｽﾀｯｸD
507C                      2372 :       [初ｺｰﾄﾞ初期化]
507C                      2373 :       RET
507C                      2374 :
507C                      2375 [初ｺｰﾄﾞ初期化]
507C 3E 01 32 A4 4F       2376         (初ｺｰﾄﾞ)=TRUE
5081 C9                   2377         RET
5082                      2378 :
5082                      2379 [PUSH_ﾏｰｸ]
5082 F5                   2380         PUSH AF
5083 2A FA 50             2381         HL=(ﾏｰｸﾎﾟｲﾝﾀ)
5086 7D D6 26 7C DE 51 D2 2382         IF HL>=式ｽﾀｯｸM_END [式ｽﾀｯｸOVER!] RET
508D A2 61
508F F1                   2383         POP  AF
5090 77 23                2384         (HL)=A    INC HL
5092 3A 6F 50 77 23       2385         (HL)=(型) INC HL
5097 22 FA 50             2386         (ﾏｰｸﾎﾟｲﾝﾀ)=HL
509A C9                   2387         RET
509B                      2388 :
509B                      2389 [POP_ﾏｰｸ]
509B CD A2 50 22 FA 50 C9 2390         [TOP_ﾏｰｸ] (ﾏｰｸﾎﾟｲﾝﾀ)=HL RET
50A2                      2391 :       HL=(ﾏｰｸﾎﾟｲﾝﾀ)
50A2                      2392 :       DEC HL  (型)=(HL)
50A2                      2393 :       DEC HL     A=(HL)
50A2                      2394 :       (ﾏｰｸﾎﾟｲﾝﾀ)=HL
50A2                      2395 :       RET
50A2                      2396 :
50A2                      2397 [TOP_ﾏｰｸ]
50A2 2A FA 50 18 05       2398         HL=(ﾏｰｸﾎﾟｲﾝﾀ) JR [SEC_ﾏｰｸ]1
50A7                      2399 :       DEC HL  (型)=(HL)
50A7                      2400 :       DEC HL     A=(HL)
50A7                      2401 :       RET
50A7                      2402 :
50A7                      2403 [SEC_ﾏｰｸ]
50A7 2A FA 50             2404         HL=(ﾏｰｸﾎﾟｲﾝﾀ)
50AA 2B                   2405         DEC HL
50AB 2B                   2406         DEC HL
50AC                      2407 [SEC_ﾏｰｸ]1
50AC 2B 7E 32 6F 50       2408         DEC HL  (型)=(HL)
50B1 2B 7E                2409         DEC HL     A=(HL)
50B3 C9                   2410         RET
50B4                      2411 :
50B4                      2412 [POP_1]
50B4 CD 9B 50             2413         [POP_ﾏｰｸ]
50B7 F5                   2414         PUSH AF
50B8 CD D0 50             2415         [POP_DATA]
50BB F1                   2416         POP  AF
50BC C9                   2417         RET
50BD                      2418 :
50BD                      2419 [PUSH_中間X]
50BD                      2420 :       HL=????
50BD 3E 00                2421         A=中間X
50BF                      2422 :       [PUSH_A&HL] RET
50BF                      2423 :
50BF                      2424 [PUSH_A&HL]
50BF E5                   2425         PUSH HL
50C0 CD 82 50             2426         [PUSH_ﾏｰｸ]
50C3 E1                   2427         POP  HL
50C4                      2428 :       [PUSH_DATA] RET
50C4                      2429 :
50C4                      2430 [PUSH_DATA]
50C4 EB                   2431         EX DE.HL
50C5 2A FC 50             2432         HL=(DATAﾎﾟｲﾝﾀ)
50C8 72 23                2433         (HL)=D  INC HL
50CA 73 23                2434         (HL)=E  INC HL
50CC 22 FC 50             2435         (DATAﾎﾟｲﾝﾀ)=HL
50CF C9                   2436         RET
50D0                      2437 :
50D0                      2438 :
50D0                      2439 [POP_DATA]
50D0 CD D8 50 ED 53 FC 50 2440         [TOP_DATA] (DATAﾎﾟｲﾝﾀ)=DE  RET
50D7 C9
50D8                      2441 :       HL=(DATAﾎﾟｲﾝﾀ)
50D8                      2442 :       DEC HL  E=(HL)
50D8                      2443 :       DEC HL  D=(HL)
50D8                      2444 :       (DATAﾎﾟｲﾝﾀ)=HL
50D8                      2445 :       EX DE.HL
50D8                      2446 :       RET
50D8                      2447 :
50D8                      2448 [TOP_DATA]
50D8 2A FC 50             2449         HL=(DATAﾎﾟｲﾝﾀ)
50DB 2B 5E                2450         DEC HL  E=(HL)
50DD 2B 56                2451         DEC HL  D=(HL)
50DF EB                   2452         EX DE.HL
50E0 C9                   2453         RET
50E1                      2454 :
50E1                      2455 [EX_STACK]
50E1 2A FA 50 CD EA 50    2456         HL=(ﾏｰｸﾎﾟｲﾝﾀ)  [EX_STACK処理]
50E7                      2457 :       [EX_DATA]
50E7                      2458 :       RET
50E7                      2459 :
50E7                      2460 [EX_DATA]
50E7 2A FC 50             2461         HL=(DATAﾎﾟｲﾝﾀ)
50EA                      2462 :       [EX_STACK処理] RET
50EA                      2463 :
50EA                      2464 [EX_STACK処理]
50EA 2B 5E                2465         DEC HL  E=(HL)
50EC 2B 56                2466         DEC HL  D=(HL)
50EE 2B 4E                2467         DEC HL  C=(HL)
50F0 2B 46                2468         DEC HL  B=(HL)
50F2 72 23                2469         (HL)=D  INC HL
50F4 73 23                2470         (HL)=E  INC HL
50F6 70 23                2471         (HL)=B  INC HL
50F8 71                   2472         (HL)=C
50F9 C9                   2473         RET
50FA                      2474 :
50FA 00 00                2475 ﾏｰｸﾎﾟｲﾝﾀ     DW $0000
50FC 00 00                2476 DATAﾎﾟｲﾝﾀ    DW $0000
50FE                      2477 :
50FE 00 00 00 00 00 00 00 2478 式ｽﾀｯｸM      DS 式ｽﾀｯｸ数*2
5105 00 00 00 00 00 00 00
510C 00 00 00 00 00 00 00
5113 00 00 00 00 00 00 00
511A 00 00 00 00 00 00 00
5121 00 00 00 00 00
5126                      2479 式ｽﾀｯｸM_END
5126 00 00 00 00 00 00 00 2480 式ｽﾀｯｸD      DS 式ｽﾀｯｸ数*2
512D 00 00 00 00 00 00 00
5134 00 00 00 00 00 00 00
513B 00 00 00 00 00 00 00
5142 00 00 00 00 00 00 00
5149 00 00 00 00 00
514E                      2481 :
514E                      2482 ;****************************************
514E                      2483 :
514E                      2484         NEXTFILE REAL-3
514E                      2484+        $CHAIN   REAL-3    03A.Asm
514E                         1 :
514E                         2 :        実数型コンパイラ  REAL
514E                         3 :
514E                         4 :          REAL-3    03A.Asm
514E                         5 :
514E                         6 :        [文] & [表]
514E                         7 :
514E                         8 ;****************************************
514E                         9 :
514E                        10 [文]
514E CD E3 39               11         [式初期化]
5151                        12 :ラベル
5151 CD 2C 3D               13         [名前?]
5154 79 FE 3A 20 07         14         IF C=":" THEN
5159 CD 4F 56               15           [ラベル]
515C CD BE 51 D8            16           [後ｶｯｺﾁｪｯｸ]  IF C RET
5160                        17         FI
5160                        18 :複合文
5160 CD CD 51 DA AA 51      19         [前ｶｯｺ] IF C [複合文] RET
5166                        20 :空文
5166 CD 23 60 3B 0D D8      21         [SPSCH] DM ";".$0D  IF C RET
516C                        22 :
516C CD 67 60               23         [TBLｻｰﾁ]
516F 49 C6 55 52            24         DM "IF"+$80      DW [IF文]
```

```
5173 43 41 53 C5 70 52      25         DM "CASE"+$80    DW [CASE文]
5179 46 4F D2 83 53         26         DM "FOR"+$80     DW [FOR文]
517E 57 48 49 4C C5 5C 55   27         DM "WHILE"+$80   DW [WHILE文]
5185 52 45 50 45 41 D4 7E   28         DM "REPEAT"+$80  DW [REPEAT文]
518C 55
518D 52 45 54 55 52 CE 1C   29         DM "RETURN"+$80  DW [RETURN文]
5194 52
5195 45 58 49 D4 9E 55      30         DM "EXIT"+$80    DW [EXIT文]
519B 47 4F 54 CF 11 56      31         DM "GOTO"+$80    DW [GOTO文]
51A1 00                     32         DM 0
51A2 38 03 21 09 52         33         IF NC THEN  HL=[式文]
51A7                        34 :
51A7 C3 81 1F               35         CALL [HL]  RET
51AA                        36 :
51AA                        37 :
51AA                        38 [複合文]
51AA                        39 :       [前ｶｯｺ]
51AA F5                     40         PUSH AF
51AB                        41         {
51AB CD F3 51 38 05         42           [後ｶｯｺ]  IF C EXIT
51B0 CD 4E 51               43           [文]
51B3 18 F6                  44         }
51B5 F1                     45         POP  AF
51B6                        46 :       [後括弧ｴﾗｰ?]
51B6                        47 :       RET
51B6                        48 :
51B6                        49 [後括弧ｴﾗｰ?]
51B6 B7 25 04               50         IF A<>0 THEN
51B9 BD C4 3E 62            51           IF A<>L [文括弧ｴﾗｰ]
51BD                        52         FI
51BD C9                     53         RET
51BE                        54 :
51BE                        55 :
51BE                        56 [後ｶｯｺﾁｪｯｸ].
51BE DD E5 CD FD 51 DD E1   57         PUSH IX  [後ｶｯｺ]  POP IX  RET
51C5 C9
51C6                        58 :
51C6                        59 :
51C6                        60 [初期値前ｶｯｺ]
51C6                        61 [BROCK前ｶｯｺ]
51C6 CD CD 51 D4 3E 62      62         [前ｶｯｺ] IF NC [文括弧ｴﾗｰ]
51CC C9                     63         RET
51CD                        64 :
51CD                        65 :
51CD                        66 [前ｶｯｺ]
51CD CD 57 60               67         [TBLｻｰﾁ]
51D0 7B 0D 01 00            68         DM "{".$0D     DW 1
51D4 5B 0D 02 00            69         DM "[".$0D     DW 2
51D8 42 45 47 49 CE 03 00   70         DM "BEGIN"+$80 DW 3
51DF 00                     71         DM 0
51E0 38 02 2E 00            72         IF NC THEN L=0
51E4 7D                     73         A=L
51E5 C9                     74         RET
51E6                        75 :
51E6                        76 :
51E6                        77 [初期値後ｶｯｺ]
51E6                        78 [BLOCK後ｶｯｺ]
51E6 CD 57 60               79         [TBLｻｰﾁ]
51E9 7D 0D 01 00            80         DM "}".$0D   DW 1
51ED 5D 0D 02 00            81         DM "]".$0D   DW 2
51F1 45 4E C4 03 00         82         DM "END"+$80 DW ENDS
51F6 00                     83         DM 0
51F7 38 02 2E 00            84         IF NC THEN L=0
51FB 7D                     85         A=L
51FC C9                     86         RET
51FD                        87 :
0003                        88 ENDS  EQU 3
51FD                        89 :
51FD                        90 [後ｶｯｺ]
51FD CD E6 51               91         [BLOCK後ｶｯｺ]
5200                        92 :
5200 F5                     93         PUSH AF
5201 FE 03 CC FE 5F         94         IF A=ENDS [ｾﾐｺﾛﾝ]
5206 F1                     95         POP  AF
5207                        96 :
5207 6F                     97         L=A
5208 C9                     98         RET
5209                        99 :
5209                       100 :
5209                       101 [式文]
5209 CD 0F 52 C3 FE 5F     102         [式文処理] [ｾﾐｺﾛﾝ] RET
520F                       103 :
520F                       104 :
520F                       105 [式文処理]
520F CD 48 3A              106         [初式]
5212 3A A6 4F FE EB 20 02  107         IF (最終ｺｰﾄﾞ)=EXS THEN DEC IY
5219 FD 2B
521B C9                    108         RET
521C                       109 :
521C                       110 :
521C                       111 [RETURN文]
521C CD AC 5B              112         [SPCUT]
521F FE 28 20 2B           113         IF A="(" THEN
5223 CD 4B 3A              114           [式]
5226 3A 1A 38 FE 01 20 13  115           IF (関数型)=VOIDS THEN
522D CD 99 62 E5 52 45 54  116             [ERROR] DM @CNT."RETURN VALUE".0
5234 55 52 4E 20 56 41 4C
523B 55 45 00
523E 18 0E                 117           ELSE
5240 CD B4 50              118             [POP_1]
5243 3A 1A 38 47 3A 6F 50  119             B=(関数型) IF (型)<>B [型ｴﾗｰ]
524A B8 C4 04 62
524E                       120           FI
524E                       121         FI
524E                       122 :
524E CD 3F 5B C9           123         @[IY]SP1 : RET
5252                       124 :
5252 C3 FE 5F              125         [ｾﾐｺﾛﾝ] RET
5255                       126 :
5255                       127 :
5255                       128 [IF文]
5255 CD 30 58 CD 40 58 FD  129         [条件式] @[IY]条件jp  PUSH IY
525C E5
525D                       130 :
525D CD 4E 51              131         [文]
5260                       132 :
5260 CD 23 60 45 4C 53 C5  133         [SPSCH] DM "ELSE"+$80
5267 30 0F                 134         IF C THEN
5269 21 00 00 CD 79 55     135           HL=0  @[IY]jp
526F                       136 :
526F D1 CD 20 58           137           POP  DE [前JP処理]
5273 FD E5                 138           PUSH IY
5275                       139 :
5275 CD 4E 51              140           [文]
5278                       141         FI
5278                       142 :
5278 D1 CD 20 58           143         POP DE [前JP処理]
527C C9                    144         RET
527D                       145 :
527D                       146 :
527D                       147 [CASE文]
527D 21 00 00 22 53 53     148         (CASEﾜｰｸ)=HL=0
5283                       149 :式
5283 3E 03 32 6F 50 CD BD  150         (型)=INTS [PUSH_中間X]
528A 50
528B                       151 :
528B CD 3B 58 CD 51 3A CD  152         [ｶｯｺ開き] [初論理式] [ｶｯｺ閉じ]
5292 36 58
5294                       153 :
5294 CD A2 50              154         [TOP_ﾏｰｸ]
5297 FE 00 20 08 CD B4 50  155         IF A=中間X : [POP_1]  @[IY]EX
529E CD A5 4F
52A1 18 03 CD 26 4C        156         ELSE       : [ｺｰﾄﾞ生成2]
52A6                       157         FI
52A6 3A 6F 50 FE 04 CC 0B  158         IF (型)=REALS [REAL不可]
52AD 62
52AE                       159 :{
52AE CD C6 51 F5           160         [BROCK前ｶｯｺ] PUSH AF
52B2                       161         {
52B2 CD F7 52 FD E5        162           CASE用[定数式] PUSH IY
52B7                       163 :文
52B7 CD 43 53              164           CASE用[文]
52BA                       165 :}
52BA CD BE 51              166           [後ｶｯｺﾁｪｯｸ]
52BD 30 06 D1 CD 20 58 18  167.          IF C THEN  POP DE [前JP処理] EXIT
52C4 24
52C5                       168 :JP ENDCASE
52C5 2A 53 53 CD 79 55     169           HL=(CASEﾜｰｸ) @[IY]jp
52CB FD E5 E1              170           HL=IY
52CE 2B 2B 22 53 53        171           DEC HL/HL  (CASEﾜｰｸ)=HL
52D3                       172 :
52D3 D1 CD 20 58           173           POP DE [前JP処理]
52D7                       174 :OTHERS:
52D7 CD 23 60 4F 54 48 45  175           [SPSCH] DM "OTHERS"+$80
52DE 52 D3
52E0 30 05 CD 43 53 18 02  176           IF C THEN  CASE用[文]  EXIT
52E7 18 C9                 177         }
52E9 CD FD 51 F1 CD B6 51  178         [後ｶｯｺ] POP AF [後括弧ｴﾗｰ?]
52F0                       179 :
52F0 ED 5B 53 53 C3 41 50  180         DE=(CASEﾜｰｸ) [ﾁｪｲﾝ処理] RET
52F7                       181 :
52F7                       182 :
52F7                       183 CASE用[定数式]
52F7 CD 22 3A              184         [INT定数式]
52FA E5 CD 23 60 54 CF E1  185         PUSH HL  [SPSCH] DM "TO"+$80  POP HL
```

▶私の周りにだんだんX68000ユーザーが増えてきた。これもOh!Xのおかげだ。

周東　正男(17)群馬県

```
5301                    186 :'A' TO 'F'
5301 30 18              187         IF C THEN
5303 CD 5A 53           188           @[IY]CPDEnn
5306 CD 44 5B 02 38 05  189           @[IY]SP DB 2 DB $38,$06
530C CD 22 3A           190           [INT定数式]
530F CD 71 53           191           @[IY]CPnnDE
5312 CD 44 5B 03 DA 00 00 192         @[IY]SP DB 3 JP C,$0000
5319                    193 :'A','C','E'
5319 18 27              194         ELSE
531B                    195           {
531B CD CE 4B           196             [PUSH_定数%HL]
531E CD 7C 50           197             [初ｺｰﾄﾞ初期化]
5321 CD D6 4B           198             [ｺｰﾄﾞ生成1]
5324 CD 44 5B 03 B7 ED 52 199           @[IY]SP DB 3 SUB HL,DE
532B                    200 :
532B CD B7 37 30 0B     201             [ｶﾝﾏ] IF NC EXIT
5330                    202 :
5330 CD 44 5B 02 28 06  203             @[IY]SP DB 2 DB $28,$06 ・
5336                    204 :
5336 CD 22 3A           205             [INT定数式]
5339 18 E0              206           }
533B CD 44 5B 03 C2 00 00 207         @[IY]SP DB 3 JP NZ,$0000
5342                    208         FI
5342 C9                 209         RET
5343                    210 ;
5343                    211 :
5343                    212 CASE用[文]
5343 CD 1D 60 3A        213         [SPINC] DB ":"
5347                    214 ;文
5347 2A 53 53 E5        215         HL=(CASEﾜｰｸ) PUSH HL
534B CD 4E 51           216         [文]
534E E1 22 53 53        217         POP HL  (CASEﾜｰｸ)=HL
5352 C9                 218         RET
5353                    219 :
5353 00 00              220 CASEﾜｰｸ DW $0000
5355                    221 :
5355                    222 :
5355                    223 @[IY]CPHLnn ;in HL:nn
5355 01 7D 7C 18 03     224         BC=$7C7D  JR @[IY]CPrrnn
535A                    225 @[IY]CPDEnn ;in HL:nn
535A 01 7B 7A           226         BC=$7A7B
535D                    227 @[IY]CPrrnn
535D EB                 228         EX DE,HL
535E 79 2E D6 63 CD 73 5B 229       A=C  L=$D6  H=E  @[IY]AHL
5365 78 2E DE 62 C3 73 5B 230       A=B  L=$DE  H=D  @[IY]AHL  RET
536C                    231 :
536C                    232 @[IY]CPnnHL ;in HL:nn
536C 01 95 9C 18 03     233         BC=$9C95  JR @[IY]CPnnrr
5371                    234 @[IY]CPnnDE ;in HL:nn
5371 01 93 9A           235         BC=$9A93
5374                    236 @[IY]CPnnrr
5374 EB                 237         EX DE,HL
5375 3E 3E 6B 61 CD 73 5B 238       A=$3E  L=E  H=C  @[IY]AHL
537C 3E 3E 6A 60 C3 73 5B 239       A=$3E  L=D  H=B  @[IY]AHL  RET
5383                    240 :
5383                    241 :
5383                    242 [FOR文]
5383 CD C2 55           243         [EXIT前処理]
5386                    244 :
5386 CD 23 60 28 0D     245         [SPSCH] DM "(",$0D
538B 30 05 CD C4 54     246         IF C : [C型FOR文]
5390 18 03 CD 98 53     247         ELSE : [P型FOR文]
5395                    248         FI
5395                    249 :
5395 C3 D5 55           250         [EXIT後処理] RET
5398                    251 :
5398                    252 :
5398                    253 [P型FOR文]
5398 CD 25 3D           254         [名前ﾁｪｯｸ]
539B CD 62 59           255         [局所表ｻｰﾁ]
539E D4 82 59           256         IF NC [大域表ｻｰﾁ]
53A1 D4 16 40           257         IF NC [未宣言変数]
53A4                    258 :
53A4 FE 02 20 08        259         IF A=変数$  THEN
53A8 78 FE 03 C4 0B 62  260           IF B<>INT$ [REAL不可]
53AE 18 0B FE 03 28 07 FE 261       EF A<>関変1$ AND A<>関変2$ THEN
53B5 04 28 03
53B8 CD F8 61           262           [名前誤用]
53BB                    263         FI
53BB 22 C1 54           264         (FOR変数DATA)=HL
53BE                    265 ;=初期値
53BE CD 1D 60 3D CD 54 3A 266       [SPINC] DB "="  [論理式]
53C5                    267 ;TO/DOWNTO
53C5 CD 67 60           268         [TBLｻｰﾁ]
53C8 54 CF 23 00        269         DM "TO"+$80     DW +$
53CC 44 4F 57 4E 54 CF 28 270       DM "DOWNTO"+$80 DW -$
53D3 00
53D4 00                 271         DM 0
53D5 38 10              272         IF NC THEN
53D7 CD 99 62 E4 54 4F 2F 273         [ERROR] DM @MIS,"TO/DOWNTO",0
53DE 44 4F 57 4E 54 4F 00
53E5 2E 23              274           L=+$
53E7                    275         FI
53E7 7D 32 C3 54        276         (TOﾃﾞｰﾀ)=L
53EB                    277 ;(変数)=HL
53EB CD C3 4B           278         [変数ｺｰﾄﾞ生成1]
53EE 3E 22 2A C1 54 CD 73 279       A=@[]HL  HL=(FOR変数DATA) @[IY]AHL
53F5 5B
53F6                    280 :
53F6 CD 79 55           281         @[IY]jp                   :JP LBL1
53F9                    282 :                                 :LOOP:
53F9 FD E5              283         PUSH IY
53FB                    284 ;++,--
53FB CD 5E 54           285         [FOR変数+-]
53FE                    286 ;終値
53FE CD 54 3A           287         [論理式]
5401 CD A2 50           288         [TOP_ﾏｰｸ]
5404 F5                 289         PUSH AF
5405 3A 6F 50 FE 04 CC 0B 290       IF (型)=REAL$ [REAL不可]
540C 62
540D F1                 291         POP  AF
540E                    292 ;中間%
540E FE 00 20 1C        293         IF A=中間% THEN
5412 CD 74 54           294           [FOR終値処理]         :比較
5415 3E D2 CD 73 5B     295           A=$D2 @[IY]AHL        :JP NC,EXIT
541A D1                 296           POP  DE
541B FD E5              297           PUSH IY
541D D5 CD 20 58        298           PUSH DE [前JP処理]    :LBL1:
5421                    299 :
5421 CD AD 54           300           P型FOR用[文]
5424                    301 :
5424 E1 CD 79 55        302           POP  HL @[IY]jp       :JP LOOP
5428 D1 CD 20 58        303           POP  DE [前JP処理]    :EXIT:
542C                    304 ;定数%/変数%
542C 18 2F              305         ELSE
542E FD E1 11 FD FF FD 19 306         POP  IY  DE=-3 ADD IY,DE
5435                    307 :                                 :LOOP:
5435 FD E5              308           PUSH IY
5437 CD B4 50           309           [POP_1]
543A E5 F5              310           PUSH HL/AF
543C 3A 6F 50 FE 04 CC 0B 311         IF (型)=REAL$ [REAL不可]
5443 62
5444                    312 :
5444 CD AD 54           313           P型FOR用[文]
5447                    314 :
5447 CD 5E 54           315           [FOR変数+-]           :INC/DEC
544A 3E 03 32 6F 50     316           (型)=INT$
544F F1 E1 CD BF 50     317           POP AF/HL [PUSH_A%HL]
5454 CD 74 54           318           [FOR終値処理]         :比較
5457 E1 3E DA CD 73 5B  319           POP HL A=$DA @[IY]AHL :JP C,LOOP
545D                    320         FI
545D                    321 :
545D C9                 322         RET
545E                    323 :
545E                    324 :
545E                    325 [FOR変数+-]
545E CD E3 39           326         [式初期化]
5461                    327 :
5461 3E 03 32 6F 50     328         (型)=INT$
5466 3E 02 2A C1 54 CD BF 329       A=変数%  HL=(FOR変数DATA) [PUSH_A%HL]
546D 50
546E                    330 :
546E 3A C3 54 C3 F3 3D  331         A=(TOﾃﾞｰﾀ) [変数+-] RET
5474                    332 :
5474                    333 :
5474                    334 [FOR終値処理]
5474                    335 ;TO
5474 3A C3 54 FE 23 20 17 336       IF (TOﾃﾞｰﾀ)=+$ THEN
547B CD A2 50           337           [TOP_ﾏｰｸ]
547E FE 01 20 08        338           IF A=定数% THEN
5482 CD B4 50 CD 55 53  339             [POP_1]      @[IY]CPHLnn
5488 18 06              340           ELSE
548A CD 26 4C CD 63 5B  341             [ｺｰﾄﾞ生成2] @[IY]SUB
5490                    342           FI
5490                    343 ;DOWNTO
5490 18 1A              344         ELSE
5492 CD A2 50           345           [TOP_ﾏｰｸ]
5495 FE 01 20 08        346           IF A=定数% THEN
5499 CD B4 50 CD 6C 53  347             [POP_1]  @[IY]CPnnHL
549F 18 0B              348           ELSE
54A1 CD 26 4C           349             [ｺｰﾄﾞ生成2]
54A4 CD 44 5B 04 7B 95 7A 350           @[IY]SP DB 4  A=E SUB L  A=D SBC H
54AB 9C
54AC                    351           FI
54AC                    352         FI
```

```
54AC                    353 :
54AC C9                 354         RET
54AD                    355 :
54AD                    356 :
54AD                    357 P型FOR用[文]
54AD 2A C1 54 E5        358         HL=(FOR変数DATA) PUSH HL
54B1 3A C3 54 F5        359         A=(TOﾃﾞｰﾀ)       PUSH AF
54B5                    360 :
54B5 CD 4E 51           361         [文]
54B8                    362 :
54B8 F1 32 C3 54        363         POP AF  (TOﾃﾞｰﾀ)=A
54BC E1 22 C1 54        364         POP HL  (FOR変数DATA)=HL
54C0 C9                 365         RET
54C1                    366 :
54C1 00 00              367 FOR変数DATA DW $0000
54C3 00                 368 TOﾃﾞｰﾀ      DB 0
54C4                    369 :
54C4                    370 :
54C4                    371 [C型FOR文]
54C4 CD 25 55 FD E5 CD 30 372       [式1] PUSH IY [式2] POP DE
54CB 55 D1
54CD                    373 :
54CD                    374 ;式2有
54CD FE 01 20 32        375         IF A=有 THEN
54D1 D5                 376           PUSH DE
54D2 CD 40 58           377             @[IY]条件jp
54D5 FD E5              378             PUSH IY
54D7 CD 79 55           379               @[IY]jp
54DA FD E5              380               PUSH IY
54DC CD 46 55           381                 [式3]
54DF C1                 382               POP BC
54E0 D1                 383             POP DE
54E1 E1                 384           POP HL
54E2                    385 :
54E2 D5                 386           PUSH DE
54E3 FE 01 20 0B        387           IF A=有 THEN          ;式3有
54E7 C5                 388             PUSH BC
54E8 C5                 389             PUSH BC
54E9                    390 :             @[IY]jmp
54E9 CD 79 55           391               @[IY]jp
54EC D1                 392             POP  DE
54ED CD 20 58           393             [前JP処理]
54F0                    394 :
54F0 18 04              395           ELSE                  ;式3無
54F2 E5                 396             PUSH HL
54F3 D5 FD E1           397             IY=DE
54F6                    398           FI
54F6 CD 4E 51           399           [文]
54F9 E1                 400           POP HL
54FA                    401 :         @[IY]jmp
54FA CD 79 55           402           @[IY]jp
54FD D1 CD 20 58        403           POP DE  [前JP処理]
5501                    404 :
5501                    405 ;式2無
5501 18 21              406         ELSE
5503 CD 79 55           407           @[IY]jp
5506 FD E5              408           PUSH IY
5508 CD 46 55           409           [式3]
550B D1                 410           POP  DE
550C FE 01 20 06        411           IF A=有 THEN          ;式3有
5510 D5                 412             PUSH DE
5511 CD 20 58           413             [前JP処理]
5514 18 07              414           ELSE                  ;式3無
5516 11 FD FF FD 19     415             DE=-3  ADD IY,DE
551B FD E5              416             PUSH IY
551D                    417           FI
551D CD 4E 51           418           [文]
5520 E1 CD 79 55        419           POP HL  @[IY]jp
5524                    420         FI
5524                    421 :
5524 C9                 422         RET
5525                    423 :
5525                    424 :
5525                    425 [式1]
5525 CD AC 5B FE 3B C4 0F 426       [SPCUT] IF A<>";" [式文処理]
552C 52
552D                    427 :
552D C3 FE 5F           428         [ｾﾐｺﾛﾝ] RET
5530                    429 :
5530                    430 [式2]
5530 CD AC 5B           431         [SPCUT]
5533 FE 3B 28 07 CD 48 3A 432       IF A<>";" : [初式] A=有
553A 3E 01
553C 18 02 3E 00        433         ELSE      :        A=無
5540                    434         FI
5540                    435 :
5540 F5 CD FE 5F F1     436         PUSH AF  [ｾﾐｺﾛﾝ]  POP AF
5545 C9                 437         RET
5546                    438 :
5546                    439 [式3]
5546 CD AC 5B           440         [SPCUT]
5549 FE 29 28 07 CD 0F 52 441       IF A<>")" : [式文処理] A=有
5550 3E 01
5552 18 02 3E 00        442         ELSE      :            A=無
5556                    443         FI
5556                    444 :
5556 F5 CD 36 58 F1     445         PUSH AF  [ｶｯｺ閉じ]  POP AF
555B C9                 446         RET
555C                    447 :
555C                    448 :
555C                    449 [WHILE文]
555C CD C2 55           450         [EXIT前処理]
555F                    451 :
555F FD E5              452         PUSH IY
5561 CD 30 58 CD 40 58  453         [条件式] @[IY]条件jp
5567 E1                 454         POP  HL
5568                    455 :
5568 FD E5              456         PUSH IY
556A E5                 457         PUSH HL
556B                    458 :
556B CD 4E 51           459         [文]
556E                    460 :
556E E1 CD 79 55        461         POP HL  @[IY]jp
5572 D1 CD 20 58        462         POP DE  [前JP処理]
5576                    463 :
5576 C3 D5 55           464         [EXIT後処理] RET
5579                    465 :
5579                    466 @[IY]jp
5579 3E C3 C3 73 5B     467         A=$C3  @[IY]AHL RET
557E                    468 :
557E                    469 :
557E                    470 [REPEAT文]
557E CD C2 55           471         [EXIT前処理]
5581                    472 :
5581 FD E5              473         PUSH IY
5583                    474 :
5583 CD 4E 51           475         [文]
5586                    476 :
5586 CD 23 60           477         [SPSCH]
5589 55 4E 54 49 CC D4 5C 478       DM "UNTIL"+$80  IF NC [UNTILなし]
5590 62
5591                    479 :
5591 CD 30 58 CD FE 5F  480         [条件式] [ｾﾐｺﾛﾝ]
5597                    481 :
5597 E1 CD 43 58        482         POP HL  @[IY]条件jpHL
559B                    483 :
559B C3 D5 55           484         [EXIT後処理] RET
559E                    485 :
559E                    486 :
559E                    487 [EXIT文]
559E 3E C3 CD 7B 5B     488         A=$C3 @[IY]A
55A3                    489 :
55A3 3A F0 55 B7 20 05  490         IF (LOOPﾚﾍﾞﾙ)=0 THEN
55A9 CD 67 62           491           [JUMP不可]
55AC                    492 :
55AC 18 11              493         ELSE
55AE CD E3 55           494           [GET_EXITﾜｰｸ]
55B1 FD E5 D1           495           DE=IY
55B4 7E 73 5F 23        496           EX E,(HL) INC HL
55B8 7E 72 57 EB CD 76 5B 497         EX D,(HL) EX DE,HL  @[IY]HL
55BF                    498         FI
55BF                    499 :
55BF C3 FE 5F           500         [ｾﾐｺﾛﾝ] RET
55C2                    501 :
55C2                    502 :
55C2                    503 [EXIT前処理]
55C2 21 F0 55 34        504         HL=LOOPﾚﾍﾞﾙ  INC (HL)
55C6                    505 :
55C6 7E FE 11 D4 AE 51  506         IF (HL)>LOOP入子上限 [入子OVER!]
55CC                    507 :
55CC CD E3 55           508         [GET_EXITﾜｰｸ]
55CF 36 00 23           509         (HL)=0  INC HL
55D2 36 00              510         (HL)=0
55D4 C9                 511         RET
55D5                    512 :
55D5                    513 :
55D5                    514 [EXIT後処理]
55D5 CD E3 55           515         [GET_EXITﾜｰｸ]
55D8 5E 23              516         E=(HL)  INC HL
55DA 56                 517         D=(HL)
55DB CD 41 50           518         [ﾁｪｲﾝ処理]
55DE                    519 :
55DE 21 F0 55 35        520         HL=LOOPﾚﾍﾞﾙ  DEC (HL)
55E2 C9                 521         RET
55E3                    522 :
```

```
55E3                    523 :
55E3                    524 [GET_EXITﾜｰｸ]
55E3 21 F1 55           525         HL=EXITﾜｰｸ
55E6 3A F0 55 3D 5F 16 00 526       A=(LOOPﾚﾍﾞﾙ) DEC A  E=A  D=0
55ED 19                 527         ADD HL,DE
55EE 19                 528         ADD HL,DE
55EF C9                 529         RET
55F0                    530 :
55F0 00                 531 LOOPﾚﾍﾞﾙ DB 0
55F1 00 00 00 00 00 00 00 532 EXITﾜｰｸ  DS LOOP入子上限*2
55F8 00 00 00 00 00 00 00
55FF 00 00 00 00 00 00 00
5606 00 00 00 00 00 00 00
560D 00 00 00 00
5611                    533 :
5611                    534 :
5611                    535 [GOTO文]
5611 3E C3 CD 7B 5B     536         A=$C3  @[IY]A
5616                    537 :
5616 CD AC 5B CD 62 59 CD 538       [SPCUT] [局所表ｻｰﾁ] @[IY]ﾗﾍﾞﾙ
561D 22 56
561F                    539 :
561F C3 FE 5F           540         [ｾﾐｺﾛﾝ] RET
5622                    541 :
5622                    542 :
5622                    543 @[IY]ﾗﾍﾞﾙ ;in A:type,HL:data,(表ﾎﾟｲﾝﾀ)
5622                    544 ;未登録
5622 B7 20 0A           545         IF A=0 THEN
5625 3E 08 CD 90 56     546           A=未ﾗﾍﾞﾙ$ [ﾗﾍﾞﾙ登録]
562A 3E 08 21 00 00     547           A=未ﾗﾍﾞﾙ$ HL=0
562F                    548         FI
562F                    549 :
562F                    550 ;未宣言
562F FE 08 20 11        551         IF A=未ﾗﾍﾞﾙ$ THEN
5633 E5                 552           PUSH HL
5634                    553 :         HL=IY  A=未ﾗﾍﾞﾙ$ [表変更]
5634 FD E5 E1 CD FC 59  554           HL=IY  [表DATA変更]
563A                    555 :
563A 3A 4E 56 3C 32 4E 56 556         INC (未ﾗﾍﾞﾙ回数)
5641 E1                 557           POP  HL
5642                    558 ;ﾗﾍﾞﾙ$以外
5642 18 07 FE 07 28 03  559         EF A<>ﾗﾍﾞﾙ$ THEN
5648 CD FB 61           560           [名前誤用]
564B                    561         FI
564B                    562 :
564B C3 76 5B           563         @[IY]HL  RET
564E                    564 :
564E 00                 565 未ﾗﾍﾞﾙ回数  DB 0
564F                    566 :
564F                    567 :
564F                    568 [ラベル]
564F CD 62 59           569         [局所表ｻｰﾁ]
5652 38 07              570         IF NC THEN
5654 3E 07 CD 90 56     571           A=ﾗﾍﾞﾙ$ [ﾗﾍﾞﾙ登録]
5659                    572 :
5659 18 30 FE 08 20 20  573         EF A=未ﾗﾍﾞﾙ$ THEN
565F E5                 574           PUSH HL
5660 FD E5 E1 3E 07 CD 0D 575         HL=IY  A=ﾗﾍﾞﾙ$  [表変更]
5667 5A
5668 E1                 576           POP  HL
5669                    577 :
5669 FD E5 C1           578           BC=IY
566C                    579           {
566C 5E 71 23           580             E=(HL) (HL)=C  INC HL
566F 56 70              581             D=(HL) (HL)=B
5671                    582 :           PUSH DE
5671 3A 4E 56 3D 32 4E 56 583           DEC (未ﾗﾍﾞﾙ回数)
5678                    584 :           POP  DE
5678 EB                 585             EX DE,HL
5679 7C B5 20 EF        586           } UNTIL HL=0
567D                    587 :
567D 18 0C FE 07 20 05  588         EF A=ﾗﾍﾞﾙ$ THEN
5683 CD F3 61           589           [二重宣言]
5686                    590 :
5686 18 03              591         ELSE
5688 CD FB 61           592           [名前誤用]
568B                    593         FI
568B                    594 :
568B 3E 3A C3 03 60     595         A=":" [INCIX] RET
5690                    596 :
5690                    597 :
5690                    598 [ﾗﾍﾞﾙ登録]
5690 F5                 599         PUSH AF
5691 CD 25 3D 3E 01 32 51 600       [名前ﾁｪｯｸ] (表域)=局所$  HL=IY
5698 5A FD E5 E1
569C F1                 601         POP  AF
569D                    602 :
569D C3 94 58           603         [表登録] RET
56A0                    604 :
56A0                    605 :
56A0                    606 [PUSH関数]
56A0                    607         {
56A0 CD 52 48           608           [引数1]処理
56A3 3A 6F 50 FE 03 20 05 609         IF (型)=INT$ THEN
56AA CD 91 4F           610             @PUSH_HL
56AD 18 06              611           ELSE
56AF 21 7B 04 CD 73 4F  612             HL=&[PUSH]-&R  @[IY]call_RT
56B5                    613           FI
56B5                    614 :
56B5 CD B7 37           615           [ｶﾝﾏ]
56B8 38 E6              616         } UNTIL NC
56BA C9                 617         RET
56BB                    618 :
56BB                    619 :
56BB                    620 [POP関数]
56BB                    621         {
56BB CD 6D 3A           622           [初論理項]
56BE CD B4 50 FE 02 C4 F8 623         [POP_1]  IF A<>変数% [名前誤用]
56C5 61
56C6                    624 :
56C6 3A 6F 50 FE 03 20 0D 625         IF (型)=INT$ THEN
56CD E5                 626             PUSH HL
56CE CD 3F 5B E1        627             @[IY]SP1 : POP HL
56D2 E1                 628             POP  HL
56D3 3E 22 CD 73 5B     629             A=@[]HL @[IY]AHL
56D8 18 0B              630           ELSE
56DA 3E 21 CD 73 5B     631             A=@HL         @[IY]AHL
56DF 21 90 04 CD 73 4F  632             HL=&[POP]-&R  @[IY]call_RT
56E5                    633           FI
56E5                    634 :
56E5 CD B7 37           635           [ｶﾝﾏ]
56E8 38 D1              636         } UNTIL NC
56EA C9                 637         RET
56EB                    638 :
56EB                    639 :
56EB                    640 [CODE関数]
56EB 3E 02 32 52 5A     641         (登録型)=CHR$
56F0                    642 :       [CODE処理]
56F0                    643 :       RET
56F0                    644 :
56F0                    645 [CODE処理] ;in (登録型), out BC:ﾊﾞｲﾄ数
56F0 FD E5              646         PUSH IY
56F2                    647         {
56F2 3A 52 5A           648           A=(登録型)
56F5 FE 03 20 05        649           IF A=INT$ THEN
56F9 CD EF 39           650             [CODE_INT]
56FC                    651 :
56FC 18 37 FE 04 20 05  652           EF A=REAL$ THEN
5702 CD F5 39           653             [CODE_REAL]
5705                    654 :
5705 18 2E              655           ELSE
5707 CD 67 60           656             [TBLｻｰﾁ]
570A 22 0D 67 57        657             DM """,$0D  DW [CODE_文字列]
570E 5B 0D 45 57        658             DM "[",$0D  DW [CODE_式]
5712 3C 0D 59 57        659             DM "<",$0D  DW [CODE_ﾗﾍﾞﾙ]
5716 28 0D 53 57        660             DM "(",$0D  DW [CODE_実数値]
571A 00                 661             DM 0
571B 30 05              662             IF C THEN
571D CD 81 1F           663               CALL [HL]
5720                    664 :
5720 18 13              665             ELSE
5722 CD 23 60 25 0D     666               [SPSCH] DM "%",$0D
5727 30 05              667               IF C THEN
5729 CD EF 39           668                 [CODE_INT]
572C 18 07              669               ELSE
572E CD 22 3A 7D CD 7B 5B 670             [INT定数式] A=L @[IY]A
5735                    671               FI
5735                    672             FI
5735                    673           FI
5735                    674 :
5735 CD B7 37           675           [ｶﾝﾏ]
5738 38 B8              676         } UNTIL NC
573A                    677 :
573A D1 FD E5 C1 79 93 4F 678       POP DE  BC=IY  SUB BC,DE
5741 78 9A 47
5744                    679 :       BC:ﾊﾞｲﾄ数
5744 C9                 680         RET
5745                    681 :
5745                    682 [CODE_式]
5745 CD 7C 50 CD 4B 3A CD 683       [初ｺｰﾄﾞ初期化] [式] [POP_1]
574C B4 50
574E                    684 :
574E 3E 5D C3 03 60     685         A="]" [INCIX]  RET
5753                    686 :
5753                    687 [CODE_実数値]
```

▶X68000を妻に乗っ取られてしまったので最近はX1をよく使用します。INTEGRAL X1によりX68000のフォーマットマシンと化していたX1。まだまだ使えると思います。

宮野　竜一(27)東京都

```
5753 CD F5 39 C3 36 58      688         [CODE_REAL] [ｶｯｺ閉じ] RET
5759                        689 :
5759                        690 [CODE_ﾗﾍﾞﾙ]
5759 CD 25 3D               691         [名前ﾁｪｯｸ]
575C CD 62 59 CD 22 56      692         [局所表ｻｰﾁ] @[IY]ﾗﾍﾞﾙ
5762                        693 :
5762 3E 3E C3 03 60         694         A=">" [INCIX]  RET
5767                        695 :
5767                        696 [CODE_文字列] ;最後に$00は付けない
5767 AF 32 42 61            697         (漢ﾌﾗｸﾞ)=0
576B                        698         {
576B DD 7E 00 FE 20 DA E4   699           A=(IX) IF A<$20 [文字列ｴﾗｰ] RET
5772 61
5773 DD 23                  700           INC IX
5775 FE 22 C8               701           IF A='"' RET
5778                        702 :
5778 CD 0F 46 CD 7B 5B      703           [文字列漢字#処理]  @[IY]A
577E 18 EB                  704         }
5780                        705 :
5780                        706 :
5780                        707 [PRINT関数]
5780                        708         {
5780 CD 89 57               709           [書式]
5783 CD B7 37               710           [ｾﾝﾌ]
5786 38 F8                  711         } UNTIL NC
5788 C9                     712         RET
5789                        713 :
5789                        714 [書式]
5789 CD 67 60               715         [ﾃｰﾌﾞﾙｻｰﾁ]
578C 53 54 52 A4 9A 01      716         DM "STR$"+$80  DW &[?STR]-&R
5792 46 4F 52 4D A4 DA 01   717         DM "FORM$"+$80 DW &[?10_n]-&R
5799 00                     718         DM 0
579A 3E 02 DA 07 58         719         A=2 IF C [書式引数] RET
579F                        720 :
579F CD 67 60               721         [ﾃｰﾌﾞﾙｻｰﾁ]
57A2 50 4E A4 C6 01         722         DM "PN$"+$80   DW &[?行号付]-&R
57A7 44 45 43 49 A4 D7 01   723         DM "DECI$"+$80 DW &[?10_5]-&R
57AE 48 45 58 32 A4 B3 01   724         DM "HEX2$"+$80 DW &[?HEX2]-&R
57B5 48 45 58 34 A4 AF 01   725         DM "HEX4$"+$80 DW &[?HEX4]-&R
57BC 4D 53 47 A4 01 02      726         DM "MSG$"+$80  DW &[?MSG]-&R
57C2 4D 53 58 A4 FD 01      727         DM "MSX$"+$80  DW &[?MSX]-&R
57C8 43 48 52 A4 A4 01      728         DM "CHR$"+$80  DW &[?CHR]-&R
57CE 52 45 41 4C A4 0C 02   729         DM "REAL$"+$80 DW &[?REAL]-&R
57D5 00                     730         DM 0
57D6 3E 01 DA 07 58         731         A=1 IF C [書式引数] RET
57DB                        732 :
57DB DD 7E 00               733         A=(IX)
57DE                        734 :"文字列"
57DE FE 22 20 0A            735         IF A='"' THEN
57E2 DD 23                  736           INC IX
57E4 CD 80 45 21 FD 01      737           [文字列処理]  HL=&[?MSX]-&R
57EA                        738 :/
57EA 18 18 FE 2F 20 0E DD   739         EF A="/" AND (IX+1)<>"/" THEN
57F1 7E 01 FE 2F 28 07
57F7 DD 23 21 A8 01         740           INC IX        HL=&[CR]-&R
57FC                        741 :式
57FC 18 06                  742         ELSE
57FE CD 52 48 21 D2 01      743           [引数1]処理  HL=&[?10]-&R
5804                        744         FI
5804 C3 73 4F               745         @[IY]call_RT  RET
5807                        746 :
5807                        747 [書式引数] ;in A,HL
5807 E5                     748         PUSH HL
5808                        749 :
5808 F5                     750         PUSH AF
5809 CD 3B 58               751         [ｶｯｺ開き]
580C F1                     752         POP  AF
580D FE 01 20 05 CD 52 48   753         IF A=1 : [引数1]処理
5814 18 03 CD 5E 48         754         ELSE   : [引数2]処理
5819                        755         FI
5819 CD 36 58               756         [ｶｯｺ閉じ]
581C                        757 :
581C E1 C3 73 4F            758         POP HL @[IY]call_RT  RET
5820                        759 :
5820                        760 :
5820                        761 [前JP処理]  ;in DE:addr
5820 FD E5 E1               762         HL=IY
5823 FD E5                  763         PUSH IY
5825                        764 :
5825 1B 1B D5 FD E1 CD 76   765         DEC DE/DE  IY=DE  @[IY]HL
582C 58
582D                        766 :
582D FD E1                  767         POP  IY
582F C9                     768         RET
5830                        769 :
5830                        770 :
5830                        771 [条件式]
5830 CD 3B 58               772         [ｶｯｺ開き]
5833 CD 48 3A               773         [初式]
5836                        774 :       [ｶｯｺ閉じ]
5836                        775 :       RET
5836                        776 :
5836                        777 [ｶｯｺ閉じ]
5836 3E 29 C3 03 60         778         A=")" [INCIX]  RET
583B                        779 :
583B                        780 [ｶｯｺ開き]
583B 3E 28 C3 03 60         781         A="(" [INCIX]  RET
5840                        782 :
5840                        783 :
5840                        784 @[IY]条件jp
5840 21 00 00               785         HL=0
5843                        786 :       @[IY]条件jpHL
5843                        787 :       RET
5843                        788 :
5843                        789 @[IY]条件jpHL
5843 E5                     790         PUSH HL
5844 CD A2 50               791         [TOP_ﾏｰｸ]
5847 3A 6F 50 FE 03 20 08   792         IF (型)=INT$ THEN
584E CD 44 5B 02 7C B5      793           @[IY]SP DB 2  A=H OR L
5854 18 06                  794         ELSE
5856 CD 44 5B 02 7E B7      795           @[IY]SP DB 2  A=(HL) OR A
585C                        796         FI
585C E1 3E CA CD 73 5B      797         POP  HL  A=Z$  @[IY]AHL
5862 C9                     798         RET
5863                        799 :
00CA                        800 Z$  EQU $CA
5863                        801 :
5863                        802 :
5863                        803 @[IY]SUB
5863 CD 44 5B 03 B7 ED 52   804         @[IY]SP DB 3  SUB HL,DE
586A C9                     805         RET
586B                        806 :
586B                        807 :**************************************
586B                        808 :
586B                        809 [大域表初期化]
586B                        810 :ﾊｯｼｭ表
586B 2A 65 5A               811         HL=(ﾊｯｼｭ表TOP)
586E ED 5B 67 5A            812         DE=(ﾊｯｼｭ表END)
5872 7C BA 20 02 7D BB 28   813         UNTIL HL=DE { A=0  #POKE+ }
5879 06 AF CD 5B 45 18 F2
5880                        814 :大域表
5880 2A 5D 5A 22 59 5A      815         (大域表ﾎﾟｲﾝﾀ)=HL=(大域表TOP)
5886                        816 :
5886 AF C3 9A 1F            817         A=0  CALL #POKE  RET
588A                        818 :
588A                        819 :
588A                        820 [局所表初期化]
588A 2A 61 5A 22 5B 5A      821         (局所表ﾎﾟｲﾝﾀ)=HL=(局所表TOP)
5890                        822 :
5890 AF C3 9A 1F            823         A=0  CALL #POKE  RET
5894                        824 :
5894                        825 :
5894                        826 [表登録]
5894                        827 ;in  (表域),A,HL,DE,(登録型)
5894                        828 ;out (表DATA),(補助DATA)
5894                        829 :
5894 F5 22 53 5A ED 53 55   830         PUSH AF  (表DATA)=HL  (補助DATA)=DE
589B 5A
589C                        831 :
589C 3A 51 5A FE 00 20 16   832         IF (表域)=大域$ THEN
58A3 CD 1D 59               833           [ﾊｯｼｭ表登録]
58A6                        834 :
58A6 2A 59 5A               835           HL=(大域表ﾎﾟｲﾝﾀ)
58A9 ED 4B 5F 5A            836           BC=(大域表END)
58AD 7D 91 7C 98 D4 83 61   837           IF HL>=BC [大域表OVER!]
58B4 11 59 5A               838           DE=大域表ﾎﾟｲﾝﾀ
58B7 18 11                  839         ELSE
58B9 2A 5B 5A               840           HL=(局所表ﾎﾟｲﾝﾀ)
58BC ED 4B 63 5A            841           BC=(局所表END)
58C0 7D 91 7C 98 D4 8A 61   842           IF HL>=BC [局所表OVER!]
58C7 11 5B 5A               843           DE=局所表ﾎﾟｲﾝﾀ
58CA                        844         FI
58CA                        845 :       HL:(大域/局所表ﾎﾟｲﾝﾀ)
58CA                        846 :       DE:大域/局所表ﾎﾟｲﾝﾀ
58CA AF 32 42 61 06 00      847         (漢ﾌﾗｸﾞ)=0  B=0
58D0                        848 :       PUSH DE
58D0                        849         {
58D0 DD 7E 00               850           A=(IX)
58D3 CD D3 60 CD E6 50 30   851           [漢TOUPPER] [英数ﾁｪｯｸ]  IF NC EXIT
58DA 0E
58DB                        852 :
58DB CD 5B 45               853           #POKE+
58DE                        854 :
58DE DD 23                  855           INC IX
58E0 04 78 FE 21 D4 67 61   856           INC B IF B>=名前最大長+1 [LONG名前!]
58E7 18 E7                  857         }
58E9 3E 0D CD 5B 45         858         A=$0D           #POKE+
58EE F1 CD 5B 45            859         POP AF          #POKE+
58F2 3A 52 5A CD 5B 45      860         A=(登録型)      #POKE+
58F8 3A 53 5A CD 5B 45      861         A=(表DATA)      #POKE+
58FE 3A 54 5A CD 5B 45      862         A=(表DATA+1)    #POKE+
5904 3A 55 5A CD 5B 45      863         A=(補助DATA)    #POKE+
590A 3A 56 5A CD 5B 45      864         A=(補助DATA+1)  #POKE+
5910 AF CD 9A 1F            865         A=0             #POKE
5914                        866 :       POP DE
5914 7D 12 13               867         (DE)=L INC DE
5917 7C 12                  868         (DE)=H
5919 22 57 5A               869         (表ﾎﾟｲﾝﾀ)=HL
591C C9                     870         RET
591D                        871 :
591D                        872 :
591D                        873 [ﾊｯｼｭ表登録]
591D CD 1C 5A               874         [ﾊｯｼｭ関数]
5920                        875         {
5920 CD 60 45 5F            876           #PEEK+  E=A
5924 CD 60 45 57 7A B3 28   877           #PEEK+  D=A  IF DE=0 EXIT
592B 1C
592C                        878 :
592C ED 5B 67 5A            879           DE=(ﾊｯｼｭ表END)
5930 7D 93 7C 9A 38 03 2A   880           IF HL>=DE THEN  HL=(ﾊｯｼｭ表TOP)
5937 65 5A
5939                        881 :
5939 ED 5B 69 5A            882           DE=(ﾊｯｼｭ関数値)
593D 7C BA 20 02 7D BB CC   883           IF HL=DE [ﾊｯｼｭ表OVER!]
5944 7C 61
5946 18 D8                  884         }
5948 E5                     885         PUSH HL
5949 2A 58 59 23            886         HL=(ﾊｯｼｭ表登録数) INC HL
594D 22 58 59               887         (ﾊｯｼｭ表登録数)=HL
5950 E1                     888         POP  HL
5951                        889 :登録
5951 ED 5B 59 5A C3 03 5A   890         DE=(大域表ﾎﾟｲﾝﾀ)  [DEC_POKE_DE] RET
5958                        891 :       DEC HL  A=D  #POKE
5958                        892 :       DEC HL  A=E  #POKE  RET
5958                        893 :
5958 00 00                  894 ﾊｯｼｭ表登録数  DW $0000
595A                        895 :
595A                        896 :
595A                        897 [表ｻｰﾁ]
595A 3A 51 5A FE 00 CA 82   898         IF (表域)=大域$ [大域表ｻｰﾁ] RET
5961 59
5962                        899 :       [局所表ｻｰﾁ]
5962                        900 :       RET
5962                        901 :
5962                        902 [局所表ｻｰﾁ]
5962                        903 ;out A:type, B:型, HL:data, (補助DATA)
5962 3E 01 32 51 5A         904         (表域)=局所$
5967                        905 :
5967 2A 61 5A               906         HL=(局所表TOP)
596A                        907         {
596A CD 94 1F B7 C8         908           #PEEK      IF A=0 RET ;表に無い
596F                        909 :
596F CD BE 59 D8            910           [表比較処理] IF C RET ;表に有った
5973                        911 :
5973                        912           {
5973 CD 60 45               913             #PEEK+
5976 FE 0D 20 F9            914           } UNTIL A=$0D
597A 23                     915           INC HL         :TYPE
597B 23                     916           INC HL         :型
597C 23 23                  917           INC HL/HL      :表DATA
597E 23 23                  918           INC HL/HL      :補助DATA
5980 18 E8                  919         }
5982                        920 :
5982                        921 [大域表ｻｰﾁ]
5982                        922 ;out A:type, B:型, HL:data, (補助DATA)
5982 3E 00 32 51 5A         923         (表域)=大域$
5987                        924 :
5987 CD 1C 5A 44 4D         925         [ﾊｯｼｭ関数] BC=HL
598C                        926         {
598C 60 69                  927           HL=BC
598E CD 60 45 5F            928           #PEEK+  E=A
5992 CD 94 1F 57 EB         929           #PEEK   D=A  EX DE,HL
5997 7C B5 28 20            930           IF HL=0 EXIT            ;表に無い
599B                        931 :
599B CD BE 59 D8            932           [表比較処理] IF C RET ;表に有った
599F                        933 :
599F ED 5B 67 5A            934           DE=(ﾊｯｼｭ表END)
59A3 03 03                  935           INC BC/BC
59A5 79 93 78 9A 38 04 ED   936           IF BC>=DE THEN  BC=(ﾊｯｼｭ表TOP)
59AC 4B 65 5A
59AF                        937 :
59AF ED 5B 69 5A            938           DE=(ﾊｯｼｭ関数値)
59B3 78 BA 20 02 79 BB 20   939         } UNTIL BC=DE
59BA D1
59BB                        940 :表に無い
59BB AF B7 C9               941         A=0  RCF  RET
59BE                        942 :
59BE                        943 :
59BE                        944 [表比較処理] ;一致しない場合 BC 保存
59BE DD E5 D1 AF 32 42 61   945         DE=IX  (漢ﾌﾗｸﾞ)=0
59C5                        946         {
59C5 CD 94 1F               947           #PEEK
59C8                        948 :
59C8 FE 0D 20 08            949           IF A=$0D THEN
59CC 1A CD E6 60 30 10      950             A=(DE)  [英数ﾁｪｯｸ]  IF NC EXIT
59D2                        951 :
59D2 B7 C9                  952             RCF  RET
59D4                        953           FI
59D4                        954 :
59D4 CD F0 59 28 02 B7 C9   955           CP漢[DE]  IF <> THEN  RCF RET
59DB                        956 :
59DB CD E6 60               957           [英数ﾁｪｯｸ]
59DE 23                     958           INC HL
59DF 13                     959           INC DE
59E0 18 E3                  960         }
59E2                        961 :一致
59E2 D5 DD E1               962         IX=DE
59E5 23                     963         INC HL               :$0D
59E6 CD 14 5B               964         [PEEK表]
59E9                        965 :       #PEEK+ C=A           :TYPE
59E9                        966 :       #PEEK+ B=A           :型
59E9                        967 :       #PEEK+ E=A           :表DATA
59E9                        968 :       #PEEK+ D=A           :表DATA
59E9                        969 :       #PEEK+ (補助DATA)=A
59E9                        970 :       #PEEK+ (補助DATA+1)=A
59E9 22 57 5A               971         (表ﾎﾟｲﾝﾀ)=HL
59EC EB                     972         EX DE,HL
59ED 79                     973         A=C
59EE                        974 :       HL=表DATA  A=TYPE  B=型  (補助DATA)
59EE 37                     975         SCF
59EF C9                     976         RET
59F0                        977 :
59F0                        978 :
59F0                        979 CP漢[DE]
59F0 C5                     980         PUSH BC
59F1                        981 :
59F1 47 1A CD D3 60 B8      982         B=A  A=(DE) [漢TOUPPER]  CP A,B
59F7                        983 :
59F7 C1                     984         POP  BC
59F8 C9                     985         RET
59F9                        986 :
59F9                        987 :
59F9                        988 [定数式表変更]
59F9 CD 22 3A               989         [INT定数式]
59FC                        990 :       [表DATA変更]
59FC                        991 :       RET
59FC                        992 :
59FC                        993 [表DATA変更]  ;in (表ﾎﾟｲﾝﾀ),HL:data
59FC EB                     994         EX DE,HL
59FD 2A 57 5A 4F            995         HL=(表ﾎﾟｲﾝﾀ) C=A
5A01 2B                     996         DEC HL                :補助DATA
5A02 2B                     997         DEC HL                :補助DATA
5A03                        998 [DEC_POKE_DE]
5A03 2B 7A CD 9A 1F         999         DEC HL  A=D  #POKE     :表DATA
5A08 2B 7B C3 9A 1F        1000         DEC HL  A=E  #POKE RET :表DATA
5A0D                       1001 :
5A0D                       1002 :
5A0D                       1003 [表変更]  ;in (表ﾎﾟｲﾝﾀ),A:type,HL:data
5A0D F5                    1004         PUSH AF
5A0E                       1005 :
5A0E CD FC 59              1006         [表DATA変更]
5A11                       1007 :
5A11 2A 57 5A 11 FA FF 19  1008         HL=(表ﾎﾟｲﾝﾀ) DE=-6 ADD HL,DE
5A18                       1009 :
5A18 F1 C3 9A 1F           1010         POP AF  #POKE  RET  :type変更
5A1C                       1011 :
5A1C                       1012 :
5A1C                       1013 [ﾊｯｼｭ関数]
5A1C DD E5 D1 01 00 00 AF  1014         DE=IX  BC=0  (漢ﾌﾗｸﾞ)=0
5A23 32 42 61
5A26                       1015         {
5A26 1A                    1016           A=(DE)
5A27 CD D3 60 CD E6 60 30  1017           [漢TOUPPER] [英数ﾁｪｯｸ] IF NC EXIT
5A2E 10
5A2F                       1018 :
5A2F 13                    1019           INC DE
5A30                       1020 :BC=(BC*9+A)*2
5A30 60 69                 1021           HL=BC
5A32 29 29 29              1022           ADD HL,HL  ADD HL,HL  ADD HL,HL
5A35 09                    1023           ADD HL,BC
5A36 4F 06 00              1024           C=A B=0
5A39 09                    1025           ADD HL,BC
5A3A 29                    1026           ADD HL,HL
5A3B 44 4D                 1027           BC=HL
5A3D 18 E7                 1028         }
5A3F 60 69 ED 5B 14 30 CD  1029         HL=BC DE=(ﾊｯｼｭ表SIZE) CALL &[剰余]
5A46 93 6A
5A48 ED 5B 65 5A 19        1030         DE=(ﾊｯｼｭ表TOP) ADD HL,DE
5A4D 22 69 5A              1031         (ﾊｯｼｭ関数値)=HL
5A50 C9                    1032         RET
5A51                       1033 :
5A51 00                    1034 表域      DB 0
5A52 00                    1035 登録型    DB 0
5A53 00 00                 1036 表DATA    DW $0000
5A55 00 00                 1037 補助DATA  DW $0000
5A57                       1038 :
5A57 00 00                 1039 表ﾎﾟｲﾝﾀ       DW $0000
5A59 00 00                 1040 大域表ﾎﾟｲﾝﾀ DW $0000
5A5B 00 00                 1041 局所表ﾎﾟｲﾝﾀ DW $0000
5A5D 00 00                 1042 大域表TOP   DW $0000
5A5F 00 00                 1043 大域表END  DW $0000
5A61 00 00                 1044 局所表TOP   DW $0000
5A63 00 00                 1045 局所表END  DW $0000
5A65 00 00                 1046 ﾊｯｼｭ表TOP   DW $0000
5A67 00 00                 1047 ﾊｯｼｭ表END  DW $0000
5A69 00 00                 1048 ﾊｯｼｭ関数値  DW $0000
5A6B                       1049 :
5A6B                       1050 :
5A6B                       1051 [未宣言LIST]  ;in (表域)
5A6B 2A 5D 5A              1052         HL=(大域表TOP)
5A6E 3A 51 5A FE 01 20 03  1053         IF (表域)=局所$ THEN HL=(局所表TOP)
5A75 2A 61 5A
5A78                       1054         {
5A78 CD 94 1F B7 C8        1055           #PEEK  IF A=0 RET
5A7D E5                    1056           PUSH HL
5A7E CD 0D 5B              1057           [LISTSUB1]
5A81 D1                    1058           POP  DE
5A82 79                    1059           A=C
5A83 FE 0A 28 08 FE 0C 28  1060           IF A=未関数$ OR A=未M関$ OR A=未ﾗﾍﾞﾙ$ THEN
5A8A 04 FE 08 20 03
5A8F CD 31 5B .            1061             [LISTSUB2]
5A92                       1062           FI
5A92                       1063 :
5A92 CD C7 1F 7E 30        1064           CALL #PAUSE  DW [HOT]
5A97 18 DF                 1065         }
5A99                       1066 :
5A99                       1067 :
5A99                       1068 [CMD_LIST]
5A99 2A 0B 5B              1069         HL=(関数表TOP)
5A9C                       1070         {
5A9C CD 94 1F B7 C8        1071           #PEEK  IF A=0 RET
5AA1 E5                    1072           PUSH HL
5AA2 CD 0D 5B              1073           [LISTSUB1]
5AA5 E5                    1074           PUSH HL
5AA6 D5                    1075             PUSH DE
5AA7                       1076 :
5AA7 79                    1077             A=C
5AA8 FE 09 38 04 3E 46     1078             IF A>=関数$  : A="F"
5AAE 18 1A FE 05 38 04 3E  1079             EF A>=配列1$ : A="A"
5AB5 41
5AB6 18 12 FE 03 38 04 3E  1080             EF A>=関変1$ : A="P"
5ABD 50
5ABE 18 0A FE 01 20 04 3E  1081             EF A=記定$   : A="C"
5AC5 43
5AC6 18 02 3E 56           1082             ELSE         : A="V"
5ACA                       1083             FI
5ACA CD F4 1F              1084             CALL #PRINT
5ACD                       1085 :
5ACD 3E 28 CD F4 1F        1086             A="(" CALL #PRINT
5AD2 78                    1087             A=B
5AD3 FE 02 20 04 3E 43     1088             IF A=CHR$   : A="C"
5AD9 18 12 FE 03 20 04 3E  1089             EF A=INT$   : A="I"
5AE0 49
5AE1 18 0A FE 04 20 04 3E  1090             EF A=REAL$  : A="R"
5AE8 52
5AE9 18 02 3E 56           1091             ELSE        : A="V"
5AED                       1092             FI
5AED CD F4 1F              1093             CALL #PRINT
5AF0 CD 1C 63 29 20 00     1094             CALL @MPRNT DM ") ",0
5AF6                       1095 :
5AF6 E1 CD BE 1F           1096             POP HL  CALL #PRTHL
5AFA 3E 3A CD F4 1F        1097             A=":"  CALL #PRINT
5AFF E1                    1098           POP HL
5B00 D1                    1099           POP DE
5B01 CD 31 5B              1100           [LISTSUB2]
5B04                       1101 :
5B04 CD C7 1F 7E 30        1102           CALL #PAUSE  DW [HOT]
5B09 18 91                 1103         }
5B0B                       1104 :
5B0B 00 00                 1105 関数表TOP DW $0000
5B0D                       1106 :
5B0D                       1107 :
5B0D                       1108 [LISTSUB1]
5B0D                       1109         {
5B0D CD 60 45              1110           #PEEK+
5B10 FE 0D 20 F9           1111         } UNTIL A=$0D
5B14                       1112 [PEEK表]
5B14 CD 60 45 4F           1113         #PEEK+ C=A   :TYPE
5B18 CD 60 45 47           1114         #PEEK+ B=A   :型
5B1C CD 60 45 5F           1115         #PEEK+ E=A   :表DATA
5B20 CD 60 45 57           1116         #PEEK+ D=A   :表DATA
5B24 CD 60 45 32 55 5A     1117         #PEEK+ (補助DATA)=A
5B2A CD 60 45 32 56 5A     1118         #PEEK+ (補助DATA+1)=A
5B30 C9                    1119         RET
5B31                       1120 :
5B31                       1121 [LISTSUB2]
5B31 E5                    1122         PUSH HL
5B32 EB                    1123         EX DE,HL
5B33                       1124         {
5B33 CD 60 45              1125           #PEEK+
5B36                       1126 :         PUSH AF
5B36 CD F4 1F              1127           CALL #PRINT
5B39                       1128 :         POP  AF
5B39 FE 0D 20 F5           1129         } UNTIL A=$0D
5B3D E1                    1130         POP  HL
5B3E C9                    1131         RET
5B3F                       1132 :
5B3F                       1133 :**************************************
5B3F                       1134 :
5B3F                       1135 @[IY]SP1
5B3F D1                    1136         POP DE
5B40 06 01 18 04           1137         B=1 JR @[IY]SP_1
5B44                       1138 :
5B44                       1139 @[IY]SP
5B44 D1                    1140         POP DE
5B45 1A 47 13              1141         B=(DE) INC DE
5B48                       1142 @[IY]SP_1
5B48                       1143         DO B {
5B48 1A 13 CD 7B 5B        1144           A=(DE) INC DE  @[IY]A
5B4D 10 F9                 1145         }
5B4F D5                    1146         PUSH DE
5B50 C9                    1147         RET
5B51                       1148 :
5B51                       1149 :
5B51                       1150 @DE[data]
5B51 21 ED 5B CD 76 5B     1151         HL=@DE[]      @[IY]HL
5B57 CD D0 50 18 1A        1152         [POP_DATA] JR @[IY]HL
5B5C                       1153 :
5B5C                       1154 @DEdata
5B5C CD D0 50              1155         [POP_DATA]
5B5F                       1156 :       @[IY]DEnn   RET
5B5F                       1157 :
5B5F                       1158 @[IY]DEnn
5B5F 3E 11 18 10           1159         A=@DE  JR @[IY]AHL
5B63                       1160 :
5B63                       1161 @HL[data]
5B63 CD D0 50 3E 2A 18 09  1162         [POP_DATA]  A=@HL[] JR @[IY]AHL
5B6A                       1163 :
5B6A                       1164 @HLdata
5B6A CD D0 50              1165         [POP_DATA]
5B6D                       1166 :       @[IY]HLnn   RET
5B6D                       1167 :
5B6D                       1168 @[IY]HLnn
5B6D 3E 21 18 02           1169         A=@HL  JR @[IY]AHL
5B71                       1170 :
5B71                       1171 @[IY]CALL
5B71 3E CD                 1172         A=$CD
5B73                       1173 :       @[IY]AHL RET
5B73                       1174 :
5B73                       1175 @[IY]AHL
5B73 CD 7B 5B              1176         @[IY]A
5B76                       1177 :       @[IY]HL  RET
5B76                       1178 :
5B76                       1179 @[IY]HL
5B76 7D CD 7B 5B           1180         A=L @[IY]A
5B7A 7C                    1181         A=H
5B7B                       1182 :       @[IY]A   RET
5B7B                       1183 :
5B7B                       1184 :
5B7B                       1185 @[IY]A
5B7B D9                    1186         EXX
5B7C                       1187 :
5B7C 08                    1188         EX AF,AF'
5B7D                       1189 :
5B7D FD E5 D1 2A C5 33 19  1190         DE=IY  HL=(OFFSET) ADD HL,DE  EX DE,HL
```

▶X68000がほしかったが学校で使うためPC-286になってしまった。しかし，まだあきらめたわけではない。いつかは……買いたい。こんな私はX1turboユーザーです。

野崎　牧人(20)神奈川県

```
5B84 EB
5B85 2A CC 33           1191          HL=(SCROBANｱﾄﾞﾚｽ)
5B88 7B 95 7A 9C 38 15  1192          IF HL=<DE THEN
5B8E                    1193 :          ADD HL,$1000
5B8E 7C C6 10 67 7B 95 7A 1194          ADD H,$10    IF DE<HL  [ﾒﾓﾘｰOVER!]
5B95 9C DC 43 61
5B99 2A 6A 1F 7B 95 7A 9C 1195          HL=(#MEMAX)  IF HL=<DE [ﾒﾓﾘｰOVER!]
5BA0 D4 43 61
5BA3                    1196          FI
5BA3                    1197 :
5BA3 08                 1198          EX AF,AF'
5BA4 12                 1199          (DE)=A
5BA5 FD 23              1200          INC IY
5BA7 CD AD 4F           1201          [最終ｺｰﾄﾞ0]
5BAA                    1202 :
5BAA D9                 1203          EXX
5BAB C9                 1204          RET
5BAC                    1205 :
5BAC                    1206 :
5BAC                    1207 [SPCUT]
5BAC                    1208          {
5BAC                    1209            {
5BAC DD 7E 00           1210              A=(IX)
5BAF                    1211 :
5BAF FE 20 20 04 DD 23  1212              IF A=" " : INC IX
5BB5 18 51 FE 0D 20 05 CD 1213            EF A=$0D : [1行取り込み]
5BBC 31 5E
5BBE                    1214 :            //ｺﾒﾝﾄ
5BBE                    1215 :            /*ｺﾒﾝﾄ*/
5BBE 18 48 FE 2F 20 19  1216              EF A="/" :
5BC4 DD 7E 01           1217                A=(IX+1)
5BC7 FE 2F 20 05 CD 31 5E 1218              IF A="/" : [1行取り込み]
5BCE 18 0B FE 2A 20 05 CD 1219              EF A="*" : [ｺﾒﾝﾄ]
5BD5 22 5C
5BD7 18 02 18 2F        1220                ELSE     : EXIT
5BDB                    1221                FI
5BDB                    1222 :            (*ｺﾒﾝﾄ*)
5BDB 18 2B FE 28 20 0C  1223              EF A="(" :
5BE1 DD 7E 01 FE 2A 20 22 1224              IF (IX+1)<>"*"  EXIT
5BE8 CD 22 5C           1225                [ｺﾒﾝﾄ]
5BEB                    1226 :            #ｺﾏﾝﾄﾞ
5BEB 18 18 FE 23 20 07  1227              EF A="#" :
5BF1 CD 6D 5C 30 14     1228                [#ｺﾏﾝﾄﾞ]  IF NC EXIT
5BF6                    1229 :
5BF6 18 10 B7 20 0B     1230              EF A=0   :
5BF8 3A 98 5D B7 28 09  1231                IF (INCL入子)=0 EXIT
5C01 CD 3E 5D           1232                [INCL終了処理]
5C04                    1233 :
5C04 18 02 18 02        1234              ELSE     :        EXIT
5C08                    1235              FI
5C08 18 A2              1236            }
5C0A 3A F7 5C FE 01 28 0D 1237          IF (#ｺﾝﾊﾟｲﾙ)=TRUE EXIT
5C11 DD 34 00 DD 35 00 28 1238          IF (IX)=0        EXIT
5C18 05
5C19                    1239 :
5C19 CD 31 5E           1240            [1行取り込み]
5C1C 18 8E              1241          }
5C1E DD 7E 00 C9        1242          A=(IX)  RET
5C22                    1243 :
5C22                    1244 :
5C22                    1245 [ｺﾒﾝﾄ]
5C22 DD 7E 00           1246          A=(IX)
5C25 FE 28 20 02 3E 29  1247          IF A="(" THEN  A=")"
5C2B                    1248 :        IF A="/" THEN  A="/"
5C2B 32 6C 5C           1249          (ｺﾒﾝﾄ記号)=A
5C2E                    1250
5C2E DD 23 DD 23        1251          INC IX  INC IX
5C32                    1252          {
5C32 DD 7E 00           1253            A=(IX)
5C35                    1254 :
5C35 FE 0D 20 05 CD 31 5E 1255          IF A=$0D : [1行取り込み]
5C3C                    1256
5C3C 18 2B B7 20 04 18 28 1257          EF A=0   : EXIT
5C43 18 24 B7 20 0B 3A 98 1258          EF A=0   : IF (INCL入子)=0  EXIT
5C4A 5D B7 28 10
5C4E CD 3E 5D           1259                       [INCL終了処理]
5C51                    1260
5C51 18 16 FE 2A 20 10 DD 1261          EF A="*" : INC IX
5C58 23
5C59 3A 6C 5C DD BE 00 20 1262                     IF (ｺﾒﾝﾄ記号)=(IX) THEN
5C60 04
5C61 DD 23 18 05        1263                         INC IX  EXIT
5C65                    1264                       FI
5C65                    1265
5C65 18 02 DD 23        1266            ELSE     : INC IX
5C69                    1267            FI
5C69 18 C7              1268          }
5C6B C9                 1269          RET
5C6C                    1270 :
5C6C 2F                 1271 ｺﾒﾝﾄ記号  DB "/"
5C6D                    1272 :
5C6D                    1273 :
5C6D                    1274 [#ｺﾏﾝﾄﾞ]
5C6D CD 6A 60           1275          [TBLｻｰﾁ2]
5C70 23 49 C6 A9 5C     1276          DM "#IF"+$80    DW [#IF]
5C75 23 45 4C 53 C5 C6 5C 1277          DM "#ELSE"+$80  DW [#ELSE]
5C7C 23 45 4E 44 49 C6 DA 1278          DM "#ENDIF"+$80 DW [#ENDIF]
5C83 5C
5C84 00                 1279          DM 0
5C85 38 1D              1280          IF NC THEN
5C87 3A F7 5C FE 00 C8  1281            IF (#ｺﾝﾊﾟｲﾙ)=FALSE RET ::RCF
5C8D CD 6A 60           1282            [TBLｻｰﾁ2]
5C90 23 43 48 41 49 CE B9 1283          DM "#CHAIN"+$80   DW [#CHAIN]
5C97 5D
5C98 23 49 4E 43 4C 55 44 1284          DM "#INCLUDE"+$80 DW [#INCLUDE]
5C9F C5 F9 5C
5CA2 00                 1285            DM 0
5CA3 D0                 1286            IF NC RET
5CA4                    1287          FI
5CA4                    1288 :
5CA4 CD 81 1F 37 C9     1289          CALL [HL]  SCF  RET
5CA9                    1290 :
5CA9                    1291 :
5CA9                    1292 [#IF]
5CA9 3A F8 5C FE 01 CC 49 1293          IF (#IFﾓｰﾄﾞ)=TRUE [#IF入子]
5CB0 62
5CB1                    1294 :
5CB1 CD 22 3A           1295          [INT定数式]
5CB4                    1296 :
5CB4 06 01 7C B5 20 02 06 1297          B=TRUE IF HL=0 THEN B=FALSE
5CBB 00
5CBC 78 32 F7 5C        1298          (#ｺﾝﾊﾟｲﾙ)=B
5CC0 3E 01 32 F8 5C     1299          (#IFﾓｰﾄﾞ)=TRUE
5CC5 C9                 1300          RET
5CC6                    1301 :
5CC6                    1302 :
5CC6                    1303 [#ELSE]
5CC6 CD E9 5C D0        1304          [#IFﾁｪｯｸ] IF NC RET
5CCA                    1305 :
5CCA 06 01 3A F7 5C FE 01 1306          B=TRUE IF (#ｺﾝﾊﾟｲﾙ)=TRUE THEN B=FALSE
5CD1 20 02 06 00
5CD5 78 32 F7 5C        1307          (#ｺﾝﾊﾟｲﾙ)=B
5CD9 C9                 1308          RET
5CDA                    1309 :
5CDA                    1310 :
5CDA                    1311 [#ENDIF]
5CDA CD E9 5C D0        1312          [#IFﾁｪｯｸ] IF NC RET
5CDE                    1313 :
5CDE 3E 01 32 F7 5C     1314          (#ｺﾝﾊﾟｲﾙ)=TRUE
5CE3 3E 00 32 F8 5C     1315          (#IFﾓｰﾄﾞ)=FALSE
5CE8 C9                 1316          RET
5CE9                    1317 :
5CE9                    1318 :
5CE9                    1319 [#IFﾁｪｯｸ]
5CE9 3A F8 5C FE 01 20 02 1320          IF (#IFﾓｰﾄﾞ)=TRUE THEN SCF RET
5CF0 37 C9
5CF2                    1321 :
5CF2 CD 53 62 B7 C9     1322          [#IFなし] RCF RET
5CF7                    1323 :
5CF7                    1324 :
5CF7 00                 1325 #ｺﾝﾊﾟｲﾙ DB 0
5CF8 00                 1326 #IFﾓｰﾄﾞ DB 0
5CF9                    1327 :
5CF9                    1328 :
5CF9                    1329 [#INCLUDE]
5CF9 3A 3B 33 FE 01 CC E4 1330          IF (ONﾒﾓﾘ)=TRUE    [INCLUDEｴﾗｰ!]
5D00 61
5D01 3A 98 5D           1331          A=(INCL入子)
5D04 FE 04 D4 B4 61     1332          IF A>=INCL入子上限 [INCLUDEｴﾗｰ!]
5D09                    1333 :
5D09 CD 88 5D           1334          [GET_INCLﾜｰｸ]
5D0C 3A 5D 1F 67        1335          H=(#DSK)
5D10 3A FB 5F 6F CD 37 5D 1336          L=(NEXTｸﾗｽﾀ)     LD[DE]HL
5D17 2A FC 5F CD 37 5D  1337          HL=(ｼｺｰﾄﾞNO)     LD[DE]HL
5D1D 2A 07 5F CD 37 5D  1338          HL=(行番号)      LD[DE]HL
5D23 2A 63 5E CD 37 5D  1339          HL=(TEXTﾎﾟｲﾝﾀ)  LD[DE]HL
5D29                    1340 :
5D29 DD E5 D1 CD 1B 5F  1341          DE=IX [OPEN]
5D2F                    1342 :
5D2F 3A 98 5D 3C 32 98 5D 1343          INC (INCL入子)
5D36 C9                 1344          RET
5D37                    1345 :
5D37                    1346 LD[DE]HL
5D37 7D 12 13           1347          (DE)=L  INC DE
5D3A 7C 12 13 C9        1348          (DE)=H  INC DE  RET
5D3E                    1349 :
5D3E                    1350 :
5D3E                    1351 [INCL終了処理]
5D3E CD 1C 63 F5 FA 0D 00 1352          CALL @MPRNT  DM @INCL.@COMPLETED,$0D,0
5D45                    1353 :
5D45 3A 98 5D 3D 32 98 5D 1354          DEC (INCL入子)
5D4C                    1355 :
5D4C CD 88 5D           1356          [GET_INCLﾜｰｸ]
5D4F CD 81 5D 7C 32 5D 1F 1357          LDHL[DE] (#DSK)=H
5D56 7D 32 FB 5F        1358                   (NEXTｸﾗｽﾀ)=L
5D5A CD 81 5D 22 FC 5F  1359          LDHL[DE] (ｼｺｰﾄﾞNO)=HL
5D60 CD 81 5D 22 07 5F  1360          LDHL[DE] (行番号)=HL
5D66 CD 81 5D 22 63 5E  1361          LDHL[DE] (TEXTﾎﾟｲﾝﾀ)=HL
5D6C                    1362 :
5D6C CD 4D 5F           1363          [FAT]
5D6F                    1364 :
5D6F ED 5B FC 5F        1365          DE=(ｼｺｰﾄﾞNO)
5D73 2A 0A 30           1366          HL=(ｸﾗｽﾀBUFF)
5D76 3E 10 CD 00 20 DC 9A 1367          A=$10  CALL #DRDSB  IF C [ABORT]
5D7D 30
5D7E                    1368 :
5D7E C3 31 5E           1369          [1行取り込み] RET
5D81                    1370 :
5D81                    1371 LDHL[DE]
5D81 1A 6F 13           1372          L=(DE)  INC DE
5D84 1A 67 13 C9        1373          H=(DE)  INC DE  RET
5D88                    1374 :
5D88                    1375 :
5D88                    1376 [GET_INCLﾜｰｸ]
5D88 11 99 5D           1377          DE=INCLﾜｰｸ
5D8B 3A 98 5D 07 07 07  1378          A=(INCL入子) RLCA RLCA RLCA  :*8
5D91 83 5F              1379          ADD E,A
5D93 7A CE 00 57        1380          ADC D,0
5D97 C9                 1381          RET
5D98                    1382 :
5D98 00                 1383 INCL入子 DB 0
5D99 00 00 00 00 00 00 00 1384 INCLﾜｰｸ  DS INCL入子上限*8
5DA0 00 00 00 00 00 00 00
5DA7 00 00 00 00 00 00 00
5DAE 00 00 00 00 00 00 00
5DB5 00 00 00 00
5DB9                    1385 :
5DB9                    1386 :
5DB9                    1387 [#CHAIN]
5DB9 3A 3B 33 FE 01 20 25 1388          IF (ONﾒﾓﾘ)=TRUE THEN
5DC0 CD C4 1F           1389            CALL #BELL
5DC3 CD E2 1F 43 48 41 49 1390          CALL #MPRNT DM "CHAIN OK ? ",0
5DCA 4E 20 4F 4B 20 3F 20
5DD1 00
5DD2 CD 21 20 FE 1B CC 7E 1391          CALL #FLGET  IF A=$1B [HOT]
5DD9 30
5DDA CD EB 1F           1392            CALL #NL
5DDD DD E5 D1 CD 5D 5F  1393            DE=IX [LOAD]
5DE3 18 06              1394          ELSE
5DE5 DD E5 D1 CD 1B 5F  1395            DE=IX [OPEN]
5DEB                    1396          FI
5DEB                    1397 :        [ｱﾄﾞﾚｽ_ﾁｪｯｸ]
5DEB                    1398 :        RET
5DEB                    1399 :
5DEB                    1400 [ｱﾄﾞﾚｽ_ﾁｪｯｸ]
5DEB FD E5 E1 ED 4B C5 33 1401          HL=IY  BC=(OFFSET)  ADD HL,BC
5DF2 09
5DF3 ED 5B 70 1F        1402          DE=(#DTADR)
5DF7 7D 93 7C 9A 38 14  1403          IF DE=<HL THEN
5DFD 2A C3 33 09        1404            HL=(OBJ先頭)            ADD HL,BC
5E01 EB                 1405            EX DE,HL
5E02 2A 70 1F ED 4B 72 1F 1406          HL=(#DTADR) BC=(#SIZE) ADD HL,BC
5E09 09
5E0A                    1407 :
5E0A 7B 95 7A 9C DC 51 61 1408          IF DE<HL [不正ｱﾄﾞﾚｽ!]
5E11                    1409          FI
5E11 C9                 1410          RET
5E12                    1411 :
5E12                    1412 :
5E12                    1413 [TEXT初期化]
5E12 2A 0A 30           1414          HL=(ｸﾗｽﾀBUFF)
5E15 3A 3B 33 FE 01 20 03 1415          IF (ONﾒﾓﾘ)=TRUE THEN  HL=(TEXT_TOP)
5E1C 2A 0C 30
5E1F 22 63 5E           1416          (TEXTﾎﾟｲﾝﾀ)=HL
5E22                    1417 :
5E22 21 00 00 22 07 5F  1418          (行番号)=HL=0
5E28                    1419 :
5E28 DD 21 65 5E DD 36 00 1420          IX=LINEBUFF  (IX)=$0D
5E2F 0D
5E30 C9                 1421          RET
5E31                    1422 :
5E31                    1423 :
5E31                    1424 [1行取り込み]
5E31 2A 63 5E 11 65 5E  1425          HL=(TEXTﾎﾟｲﾝﾀ)  DE=LINEBUFF
5E37 06 00              1426          B=0
5E39                    1427          {
5E39 04 78 FE 81 D4 5D 61 1428            INC B  IF B>LINEBUFF長 [LONG行!]
5E40 CD A3 5F           1429            [GETBYTE]
5E43 12 B7 28 05        1430            (DE)=A  IF A=0 EXIT
5E47 13                 1431            INC DE
5E48 FE 0D 20 ED        1432          } UNTIL A=$0D
5E4C                    1433 :
5E4C 22 63 5E           1434          (TEXTﾎﾟｲﾝﾀ)=HL
5E4F                    1435 :
5E4F 2A 07 5F 23 22 07 5F 1436          HL=(行番号) INC HL (行番号)=HL
5E56                    1437 :
5E56 3A 06 5F FE 01 CC E5 1438          IF (ﾘｽﾄ表示SW)=ON [行表示]
5E5D 5E
5E5E                    1439 :
5E5E DD 21 65 5E        1440          IX=LINEBUFF
5E62 C9                 1441          RET
5E63                    1442 :
5E63 00 00              1443 TEXTﾎﾟｲﾝﾀ   DW $0000
5E65 00 00 00 00 00 00 00 1444 LINEBUFF    DS  LINEBUFF長
5E6C 00 00 00 00 00 00 00
5E73 00 00 00 00 00 00 00
5E7A 00 00 00 00 00 00 00
5E81 00 00 00 00 00 00 00
5E88 00 00 00 00 00 00 00
5E8F 00 00 00 00 00 00 00
5E96 00 00 00 00 00 00 00
5E9D 00 00 00 00 00 00 00
5EA4 00 00 00 00 00 00 00
5EAB 00 00 00 00 00 00 00
5EB2 00 00 00 00 00 00 00
5EB9 00 00 00 00 00 00 00
5EC0 00 00 00 00 00 00 00
5EC7 00 00 00 00 00 00 00
5ECE 00 00 00 00 00 00 00
5ED5 00 00 00 00 00 00 00
5EDC 00 00 00 00 00 00 00
5EE3 00 00
5EE5                    1445 :
5EE5                    1446 :
5EE5                    1447 [行表示]
5EE5 3A 65 5E B7 28 15  1448          IF (LINEBUFF)<>0 THEN
5EEB FD E5 E1 CD BE 1F  1449            HL=IY          CALL #PRTHL
5EF1 CD 09 5F CD F1 1F  1450            [行番号表示] CALL #PRNTS
5EF7 11 65 5E CD E8 1F CD 1451          DE=LINEBUFF  CALL #MSG    CALL #NL
5EFE EB 1F
5F00                    1452          FI
5F00                    1453 :
5F00 CD C7 1F 7E 30     1454          CALL #PAUSE  DW [HOT]
5F05 C9                 1455          RET
5F06                    1456 :
5F06 00                 1457 ﾘｽﾄ表示SW DB 0
5F07 00 00              1458 行番号     DW $0000
5F09                    1459 :
5F09                    1460 :
5F09                    1461 [行番号表示]
5F09 2A 07 5F           1462          HL=(行番号)
5F0C                    1463 :        [PRT10]
5F0C                    1464 :        RET
5F0C                    1465 :
5F0C                    1466 [PRT10]
5F0C 11 15 5F CD 83 68  1467          DE=WORK10進化  &[10進化]
5F12                    1468 :        DE=WORK10進化
5F12 C3 E5 1F           1469          CALL #MSX  RET
5F15                    1470 :
5F15 31 32 33 34 35 00  1471 WORK10進化  DM "12345",0
5F1B                    1472 :
5F1B                    1473 :        for DISK
5F1B                    1474 :
5F1B                    1475 [OPEN]
5F1B 3E 04              1476          A=4
5F1D CD A3 1F           1477          CALL #FILE
5F20 CD 09 20 DC 9A 30  1478          CALL #ROPEN  IF C [ABORT]
5F26 CD 4D 5F           1479          [FAT]
5F29 CD 91 5F           1480          [Loading]
5F2C                    1481 :
5F2C 2A 74 1F 7D C6 1E 6F 1482          HL=(#IBFAD) ADD HL,$1E (NEXTｸﾗｽﾀ)=(HL)
5F33 7C CE 00 67 7E 32 FB
5F3A 5F
5F3B CD C8 5F           1483          [READｸﾗｽﾀ]
5F3E                    1484 :
5F3E                    1485 :for [ｱﾄﾞﾚｽ_ﾁｪｯｸ]
5F3E 2A 0A 30 22 70 1F  1486          (#DTADR)=HL=(ｸﾗｽﾀBUFF)
5F44 21 00 10 22 72 1F  1487          (#SIZE)=HL=$1000
5F4A                    1488 :
5F4A C3 12 5E           1489          [TEXT初期化]  RET
5F4D                    1490 :
5F4D                    1491 :
5F4D                    1492 [FAT]
5F4D ED 5B 5E 1F        1493          DE=(#FATPOS)
5F51 2A 62 1F           1494          HL=(#FATBF)
5F54 3E 01 CD 00 20 DC 9A 1495          A=1  CALL #DRDSB  IF C [ABORT]
5F5B 30
5F5C C9                 1496          RET
5F5D                    1497 :
5F5D                    1498 :
5F5D                    1499 [LOAD]
5F5D 3E 04 CD A3 1F     1500          A=4 CALL #FILE
5F62                    1501          {
5F62 CD 09 20 DC 9A 30  1502            CALL #ROPEN  IF C [ABORT]
5F68 28 14              1503                         IF Z EXIT
5F6A                    1504 :
5F6A CD E2 1F 46 4F 55 4E 1505          CALL #MPRNT  DM "FOUND   ",0
5F71 44 20 20 20 00
5F76 CD 9D 1F           1506            CALL #FPRNT
5F79 CD EB 1F           1507            CALL #NL
5F7C 18 E4              1508          }
5F7E CD 91 5F           1509          [Loading]
5F81                    1510 :
5F81 2A 0C 30 22 70 1F  1511          (#DTADR)=HL=(TEXT_TOP)
5F87                    1512 :
5F87 CD 12 5E           1513          [TEXT初期化]
5F8A                    1514 :
5F8A CD A6 1F DC 9A 30  1515          CALL #RDD  IF C [ABORT]
5F90 C9                 1516          RET
5F91                    1517 :
5F91                    1518 :
5F91                    1519 [Loading]
5F91 CD E2 1F 4C 4F 41 44 1520          CALL #MPRNT  DM "LOADING ",0
5F98 49 4E 47 20 00
5F9D CD 9D 1F           1521          CALL #FPRNT
5FA0 C3 EB 1F           1522          CALL #NL     RET
5FA3                    1523 :
5FA3                    1524 :
5FA3                    1525 [GETBYTE] :in HL  out HL
5FA3 3A 3B 33 FE 00 20 1E 1526          IF (ONﾒﾓﾘ)=FALSE THEN  :DISKから読む
5FAA C5 D5              1527            PUSH BC/DE
5FAC ED 5B 0A 30 7B C6 00 1528          DE=(ｸﾗｽﾀBUFF)  ADD DE,$1000
5FB3 5F 7A CE 10 57
5FB8 7C BA 20 02 7D BB 20 1529          IF HL=DE THEN
5FBF 05
5FC0 CD C8 5F 2A 0A 30  1530              [READｸﾗｽﾀ]  HL=(ｸﾗｽﾀBUFF)
5FC6                    1531            FI
5FC6 D1 C1              1532            POP  DE/BC
5FC8                    1533          FI
5FC8                    1534 :
5FC8 7E 23              1535          A=(HL) INC HL
5FCA C9                 1536          RET
5FCB                    1537 :
5FCB                    1538 :
5FCB                    1539 [READｸﾗｽﾀ]
5FCB 2A 62 1F           1540          HL=(#FATBF)
5FCE 3A FB 5F 5F 16 00 19 1541          E=(NEXTｸﾗｽﾀ) D=0  ADD HL,DE
5FD5                    1542 :
5FD5 7E                 1543          A=(HL)
5FD6 B7 20 05 3E 07 CD 9A 1544          IF A=0    THEN A=7  [ABORT]
5FDD 30
5FDE FE 80 38 01 AF     1545          IF A>=$80 THEN A=0
5FE3 32 FB 5F           1546          (NEXTｸﾗｽﾀ)=A
5FE6                    1547 :DE:ｸﾗｽﾀNO -> ｼｺｰﾄﾞNO
5FE6 EB                 1548          EX DE,HL
5FE7 29 29              1549          ADD HL,HL  ADD HL,HL
5FE9 29 29 22 FC 5F     1550          ADD HL,HL  ADD HL,HL  (ｼｺｰﾄﾞNO)=HL
5FEE EB                 1551          EX DE,HL
5FEF                    1552 :
5FEF 2A 0A 30           1553          HL=(ｸﾗｽﾀBUFF)
5FF2 3E 10 CD 00 20 DC 9A 1554          A=$10  CALL #DRDSB  IF C [ABORT]
5FF9 30
5FFA C9                 1555          RET
5FFB                    1556 :
5FFB 00                 1557 NEXTｸﾗｽﾀ  DB 0
5FFC 00 00              1558 ｼｺｰﾄﾞNO   DW $0000
5FFE                    1559 :
5FFE                    1560 :
5FFE                    1561 [ｾﾐｺﾛﾝ]
5FFE CD AC 5B           1562          [SPCUT]
6001 3E 3B              1563          A=";"
6003                    1564 :        [INCIX] RET
6003                    1565 :
6003                    1566 :
6003                    1567 [INCIX] ;in A
6003 F5 CD AC 5B F1     1568          PUSH AF  [SPCUT]  POP AF
6008                    1569 :        [NOSPC_INCIX]
6008                    1570 :        RET
6008                    1571 :
6008                    1572 [NOSPC_INCIX]
6008 DD BE 00 20 04     1573          IF A=(IX) THEN
600D DD 23              1574            INC IX
600F 18 0B              1575          ELSE
6011 32 19 60           1576            (INCIXｴﾗｰCHAR)=A
6014 CD 99 62 E4 5B     1577            [ERROR]  DM @MIS,"["
6019                    1578 INCIXｴﾗｰCHAR
6019 3F 5D 00           1579            DB "?]",0
601C                    1580          FI
601C C9                 1581          RET
601D                    1582 :
601D                    1583 :
601D                    1584 [SPINC]
601D D1 1A 13 D5        1585          POP DE  A=(DE) INC DE PUSH DE
6021                    1586 :
6021 18 E0              1587          JR [INCIX]
6023                    1588 :
6023                    1589 :
6023                    1590 [SPSCH]          ;漢字名は不可
6023 CD AC 5B           1591          [SPCUT]
6026 DD E5 E1           1592          HL=IX
6029 D1                 1593          POP DE
602A                    1594          {
602A 1A 13              1595            A=(DE)  INC DE
602C                    1596 :
602C FE 0D 28 31        1597            IF A=$0D JR SPSCH.OK
6030                    1598 :
6030 FE 80 38 14        1599            IF A>=$80  THEN
6034 D6 80              1600              SUB $80
6036 CD CC 60 20 23     1601              CP[HL]     IF <> EXIT :不一致
603B AF 32 42 61        1602              (漢ﾌﾗｸﾞ)=0
603F 23                 1603              INC HL
6040 7E                 1604              A=(HL)
6041 CD E6 60 38 18     1605              [英数ﾁｪｯｸ] IF C  EXIT :不一致
6046 18 19              1606              JR SPSCH.OK
6048                    1607            FI
6048                    1608 :
6048 CD CC 60 28 0E     1609            CP[HL]  IF <> THEN
604D                    1610              {
604D 1A 13              1611                A=(DE)  INC DE
604F FE 20 38 06        1612                IF A<$20  EXIT :0 or $0D
6053 FE 80 30 02        1613                IF A>=$80 EXIT
6057 18 F4              1614              }
6059 18 03              1615              EXIT :不一致
605B                    1616            FI
605B 23                 1617            INC HL
605C 18 CC              1618          }
605E                    1619 :不一致
605E B7 D5 C9           1620          RCF  PUSH DE  RET
6061                    1621 :
6061                    1622 :一致
6061                    1623 SPSCH.OK
6061 E5 DD E1           1624          IX=HL
6064 37 D5 C9           1625          SCF  PUSH DE  RET
6067                    1626 :
6067                    1627 :
6067                    1628 [TBLｻｰﾁ]          :漢字名は不可
6067                    1629 :
6067 CD AC 5B           1630          [SPCUT]
606A                    1631 :        [TBLｻｰﾁ2] RET
606A                    1632 :
606A                    1633 [TBLｻｰﾁ2]
606A                    1634 :
606A                    1635 :        漢字名は不可
606A                    1636 :
606A                    1637 :        一致したらＨＬにデータを返す
606A                    1638 :
606A D1                 1639          POP DE
606B                    1640          {
606B DD E5 E1           1641            HL=IX
606E                    1642            {
606E 1A                 1643              A=(DE)
606F FE 0D 28 37        1644              IF A=$0D  JR TBLｻｰﾁ.OK
6073 FE 80 38 14        1645              IF A>=$80 THEN
6077 D6 80              1646                SUB $80
6079 CD CC 60 20 0B     1647                CP[HL]  IF = THEN
607E AF 32 42 61        1648                  (漢ﾌﾗｸﾞ)=0
6082 23                 1649                  INC HL
6083 7E                 1650                  A=(HL)
6084 CD E6 60 30 21     1651                  [英数ﾁｪｯｸ]  IF NC JR TBLｻｰﾁ.OK
6089                    1652                FI
6089 18 09              1653                EXIT :不一致
608B                    1654              FI
608B CD CC 60 20 04     1655              CP[HL]  IF <> EXIT :不一致
6090 23                 1656              INC HL
6091 13                 1657              INC DE
```

▶X68000XVIもいいけど，私は廉価なX68000がほしい。拡張スロットは1つでいいし，ハードディスクも内蔵できなくていいから，198,000円ぐらいのマシンが出てくれたら私もX68000ユーザーになれるのに。
宮城　義和(28)静岡県

```
6092 18 DA                 1658                }
6094                       1659 :              不一致
6094                       1660                {
6094 1A 13                 1661                  A=(DE)  INC DE
6095 FE 0D 28 06           1662                  IF A=$0D  EXIT
609A FE 80 30 02           1663                  IF A>=$80 EXIT
609E 18 F4                 1664                }
60A0 13                    1665                INC DE
60A1 13                    1666                INC DE
60A2 1A B7 20 C5           1667              } UNTIL (DE)=0
60A6                       1668 :            不一致
60A6 13                    1669              INC DE
60A7 B7 D5 C9              1670              RCF  PUSH DE  RET
60AA                       1671
60AA                       1672 :一致
60AA                       1673 TBLｻｰﾁ.OK
60AA E5 DD E1              1674              IX=HL
60AD 13                    1675              INC DE
60AE 1A 6F 13              1676              L=(DE)  INC DE
60B1 1A 67 13              1677              H=(DE)  INC DE
60B4                       1678 :
60B4 1A B7 28 10           1679              UNTIL (DE)=0 {
60B8                       1680                {
60B8 1A 13                 1681                  A=(DE)  INC DE
60BA FE 0D 28 06           1682                  IF A=$0D  EXIT
60BE FE 80 30 02           1683                  IF A>=$80 EXIT
60C2 18 F4                 1684                }
60C4 13                    1685                INC DE
60C5 13                    1686                INC DE
60C6 18 EC                 1687              }
60C8                       1688 :HL:DATA
60C8 13                    1689              INC DE
60C9 37 D5 C9              1690              SCF  PUSH DE  RET
60CC                       1691 :
60CC                       1692 :
60CC                       1693 CP[HL]
60CC                       1694 :            PUSH BC
60CC 47                    1695              B=A
60CD 7E CD DD 60 B8        1696              A=(HL)  [TOUPPER]  CP A,B
60D2                       1697 :            POP BC
60D2 C9                    1698              RET
60D3                       1699 :
60D3                       1700 :
60D3                       1701 [漢TOUPPER]
60D3 08                    1702              EX AF,AF'
60D4                       1703 :
60D4 3A 42 61 B7 28 02 08  1704              IF (漢ﾌﾗｸﾞ)<>0 THEN  EX AF,AF'  RET
60DB C9
60DC                       1705 :
60DC 08                    1706              EX AF,AF'
60DD                       1707 :            [TOUPPER]
60DD                       1708 :            RET
60DD                       1709 :
60DD                       1710 [TOUPPER]
60DD FE 61 D8              1711              IF A<"a"    RET
60E0 FE 7B D0              1712              IF A>="z"+1 RET
60E3                       1713 :
60E3 C6 E0 C9              1714              ADD A,"A"-"a"  RET
60E5                       1715 :
60E6                       1716 :
60E6                       1717 [英数ﾁｪｯｸ]
60E6                       1718 :            "0"-"9" "A"-"Z" : CY
60E6 CD F1 60 D8           1719              [英字ﾁｪｯｸ]  IF C RET
60EA                       1720 :            [数字ﾁｪｯｸ]
60EA                       1721 :            RET
60EA                       1722 :
60EA                       1723 [数字ﾁｪｯｸ]
60EA                       1724 :            "0"-"9" : CY
60EA FE 3A D0              1725              CP A,"9"+1  IF NC RET
60ED FE 30 3F C9           1726              CP A,"0"    CCF  RET
60F1                       1727 :
60F1                       1728 [英字ﾁｪｯｸ]
60F1                       1729 :CY : A-Z,a-z,@,"
60F1 08                    1730              EX AF,AF'
60F2 3A 42 61              1731              A=(漢ﾌﾗｸﾞ)
60F5 B7 28 03              1732              IF A<>0 THEN
60F8 AF 32 42 61           1733                (漢ﾌﾗｸﾞ)=0
60FC 08                    1734                EX AF,AF'
60FD FE 40 38 3F           1735                IF A<$40 JR 英字NG
6101                       1736 :              JR 英字OK
6101 18 3B                 1737              ELSE
6103 08                    1738                EX AF,AF'
6104 FE 40 38 38           1739                IF A<$40 JR 英字NG
6108 FE 5B 38 32           1740                IF A<"[" JR 英字OK  :@A-Z
610C FE 5E 38 30           1741                IF A<"^" JR 英字NG
6110 FE 7B 28 2C           1742                IF A="{" JR 英字NG
6114 FE 7D 28 28           1743                IF A="}" JR 英字NG
6118 FE 80 38 22           1744                IF A<$80 JR 英字OK  :^_a-z
611C FE A0 30 09           1745                IF A<$A0  THEN
6120 08 3E 01 32 42 61 08  1746                  EX AF,AF'  (漢ﾌﾗｸﾞ)=1  EX AF,AF'
6127 18 15                 1747                  JR 英字OK
6129                       1748                FI
6129 FE E0 38 09           1749                IF A>=$E0 THEN
612D 08 3E 01 32 42 61 08  1750                  EX AF,AF'  (漢ﾌﾗｸﾞ)=1  EX AF,AF'
6134 18 08                 1751                  JR 英字OK
6136                       1752                FI
6136 FE A2 28 06           1753                IF A="｢" JR 英字NG
613A FE A3 28 02           1754                IF A="｣" JR 英字NG
613E                       1755 :              JR 英字OK
613E                       1756              FI
613E                       1757 英字OK
613E 37 C9                 1758              SCF  RET
6140                       1759 :
6140                       1760 英字NG
6140 B7 C9                 1761              RCF  RET
6142                       1762
6142 00                    1763 漢ﾌﾗｸﾞ DB $00
6143                       1764 :
6143                       1765 :=====================================
6143                       1766 :
6143                       1767              NEXTFILE REAL-4
6143                       1767+             SCHAIN   REAL-4    03A.Asm
6143                          1 :
6143                          2 :             実数型コンパイラ　REAL
6143                          3 :
6143                          4 :               REAL-4    03A.Asm
6143                          5 :
6143                          6 :             [ERROR] & [ﾗﾝﾀｲﾑ･ﾙｰﾁﾝ]
6143                          7 :
6143                          8 :=====================================
6143                          9 :
6143                         10 [ﾒﾓﾘｰOVER!]
6143 CD 8C 62 4F 55 54 E8    11         [最悪ERROR] DM "OUT",@OF,"MEMORY",0
614A 4D 45 4D 4F 52 59 00
6151                         12 :
6151                         13 [不正ｱﾄﾞﾚｽ!]
6151 CD 8C 62 E1 41 44 44    14         [最悪ERROR] DM @ILL,"ADDRESS",0
6158 52 45 53 53 00
615D                         15 :
615D                         16 [LONG行!]
615D CD 8C 62 E6 EA 4C 49    17         [最悪ERROR] DM @TOO,@LNG,"LINE",0
6164 4E 45 00
6167                         18 :
6167                         19 [LONG名前!]
6167 CD 8C 62 E6 EA EE 00    20         [最悪ERROR] DM @TOO,@LNG,@NME,0
616E                         21 :
616E                         22 [関数WKOVER!]
616E CD 8C 62 F0 ED E3 00    23         [最悪ERROR] DM @FUNC,@TBL,@OVF,0
6175                         24 :
6175                         25 [定数表OVER!]
6175 CD 8C 62 EF ED E3 00    26         [最悪ERROR] DM @CONST,@TBL,@OVF,0
617C                         27 :
617C                         28 [ﾊｯｼｭ表OVER!]
617C CD 8C 62 F9 ED E3 00    29         [最悪ERROR] DM @HASH,@TBL,@OVF,0
6183                         30 :
6183                         31 [大域表OVER!]
6183 CD 8C 62 F6 ED E3 00    32         [最悪ERROR] DM @GLBL,@TBL,@OVF,0
618A                         33 :
618A                         34 [局所表OVER!]
618A CD 8C 62 F8 ED E3 00    35         [最悪ERROR] DM @LOCAL,@TBL,@OVF,0
6191                         36 :
6191                         37 [TEMP_OVER!]
6191 CD 8C 62 54 45 4D 50    38         [最悪ERROR] DM "TEMP",@TBL,@OVF,0
6198 ED E3 00
619B                         39 :
619B                         40 [定数ｽﾀｯｸOVER!]
619B CD 8C 62 EF F4 E3 00    41         [最悪ERROR] DM @CONST,@STACK,@OVF,0
61A2                         42 :
61A2                         43 [式ｽﾀｯｸOVER!]
61A2 CD 8C 62 E6 43 4F 4D    44         [最悪ERROR] DM @TOO,"COMPLEX",0
61A9 50 4C 45 58 00
61AE                         45 :
61AE                         46 [入子OVER!]
61AE CD 8C 62 F2 E3 00       47         [最悪ERROR] DM @NEST,@OVF,0
61B4                         48 :
61B4                         49 [INCLUDEｴﾗｰ!]
61B4 CD 8C 62 E5 F5 00       50         [最悪ERROR] DM @CNT,@INCL,0
61BA                         51 :
61BA                         52 [REAL不可!]
61BA CD 8C 62 E1 F7 E8 FB    53         [最悪ERROR] DM @ILL,@USE,@OF,@REAL,0
61C1 00
61C2                         54 :
61C2                         55 [文法ｴﾗｰ!]
61C2 CD 8C 62 F1 E0 00       56         [最悪ERROR] DM @SYNTAX,@ERR,0
61C8                         57 :
```

```
61C8                         58 [二重宣言!]
61C8 CD 8C 62 FC E8 EE 00    59         [最悪ERROR] DM @REDEF,@OF,@NME,0
61CF                         60 :
61CF                         61 [引数過多!]
61CF CD 8C 62 E6 F3 E2 00    62         [最悪ERROR] DM @TOO,@MANY,@ARG,0
61D6                         63 :
61D6                         64 :
61D6                         65 [文法ｴﾗｰ]
61D6 CD 99 62 F1 E0 00       66         [ERROR] DM @SYNTAX,@ERR,0
61DC C9                      67         RET
61DD                         68 :
61DD                         69 [定数ｴﾗｰ]
61DD CD 99 62 E1 EF 00       70         [ERROR] DM @ILL,@CONST,0
61E3 C9                      71         RET
61E4                         72 :
61E4                         73 [文字列ｴﾗｰ]
61E4 CD 99 62 E1 EB 00       74         [ERROR] DM @ILL,@STR,0
61EA C9                      75         RET
61EB                         76 :
61EB                         77 [LONG文字列]
61EB CD 99 62 E6 EA EB 00    78         [ERROR] DM @TOO,@LNG,@STR,0
61F2 C9                      79         RET
61F3                         80 :
61F3                         81 [二重宣言]
61F3 CD 99 62 FC E8 EE 00    82         [ERROR] DM @REDEF,@OF,@NME,0
61FA C9                      83         RET
61FB                         84 :
61FB                         85 [名前誤用]
61FB CD 99 62 E1 F7 E8 EE    86         [ERROR] DM @ILL,@USE,@OF,@NME,0
6202 00
6203 C9                      87         RET
6204                         88 :
6204                         89 [型ｴﾗｰ]
6204 CD 99 62 E1 E9 00       90         [ERROR] DM @ILL,@TYP,0
620A C9                      91         RET
620B                         92 :
620B                         93 [REAL不可]
620B CD 99 62 E1 F7 E8 FB    94         [ERROR] DM @ILL,@USE,@OF,@REAL,0
6212 00
6213 C9                      95         RET
6214                         96 :
6214                         97 [DATA過多]
6214 CD 99 62 E6 F3 44 41    98         [ERROR] DM @TOO,@MANY,"DATA",0
621B 54 41 00
621E C9                      99         RET
621F                        100 :
621F                        101 [引数過多]
621F CD 99 62 E6 F3 E2 00   102         [ERROR] DM @TOO,@MANY,@ARG,0
6226 C9                     103         RET
6227                        104 :
6227                        105 [引数数ｴﾗｰ]
6227 CD 99 62               106         [ERROR]
622A E7 4E 55 4D 42 45 52   107         DM @MMATCH,"NUMBER",@OF,@ARG,0
6231 E8 E2 00
6234 C9                     108         RET
6235                        109 :
6235                        110 [引数型ｴﾗｰ]
6235 CD 99 62 E7 E9 E8 E2   111         [ERROR] DM @MMATCH,@TYP,@OF,@ARG,0
623C 00
623D C9                     112         RET
623E                        113 :
623E                        114 [文括弧ｴﾗｰ]
623E CD 99 62 E1 42 52 41   115         [ERROR] DM @ILL,"BRACE",0
6245 43 45 00
6248 C9                     116         RET
6249                        117 :
6249                        118 [#IF入子]
6249 CD 99 62 23 49 46 20   119         [ERROR] DM "#IF ",@NEST,0
6250 F2 00
6252 C9                     120         RET
6253                        121 :
6253                        122 [#IFなし]
6253 CD 99 62 E4 23 49 46   123         [ERROR] DM @MIS,"#IF",0
625A 00
625B C9                     124         RET
625C                        125 :
625C                        126 [UNTILなし]
625C CD 99 62 E4 55 4E 54   127         [ERROR] DM @MIS,"UNTIL",0
6263 49 4C 00
6266 C9                     128         RET
6267                        129 :
6267                        130 [JUMP不可]
6267 CD 99 62 E5 4A 55 4D   131         [ERROR] DM @CNT,"JUMP",0
626E 50 00
6270 C9                     132         RET
6271                        133 :
6271                        134 [範囲OVER]
6271 CD 99 62 4F 55 54 E8   135         [ERROR] DM "OUT",@OF,"RANGE",0
6278 52 41 4E 47 45 00
627E C9                     136         RET
627F                        137 :
627F                        138 [0除算]
627F CD 99 62 44 49 56 20   139         [ERROR] DM "DIV BY 0",0
6286 42 59 20 30 00
628B C9                     140         RET
628C                        141 :
628C                        142 :
628C                        143 [最悪ERROR] :終了
628C CD E2 62 CD E2 1F 20   144         [ERROR処理]  CALL #MPRNT DM " !",$0D,0
6293 21 0D 00
6296 C3 7E 30               145         JP [HOT]
6299                        146 :
6299                        147 [ERROR]      :ｷｰ入力待ち
6299 CD E2 62 CD A8 62 C3   148         [ERROR処理]  [HITKEY]  CALL #LTNL  RET
62A0 EE 1F
62A2                        149 :
62A2                        150 [ERROR2]     :ｷｰ入力なし
62A2 CD E2 62 C3 EE 1F      151         [ERROR処理]             CALL #LTNL  RET
62A8                        152 :
62A8                        153 :
62A8                        154 [HITKEY]
62A8 CD B3 62               155         [NOKEY]
62AB CD 21 20 FE 1B CA 7E   156         CALL #FLGET  IF A=$1B JP [HOT]
62B2 30
62B3                        157 :       [NOKEY]
62B3                        158 :       RET
62B3                        159 :
62B3                        160 [NOKEY]
62B3 F5                     161         PUSH AF
62B4 CD D0 1F B7 20 FA      162         { CALL #GETKY } UNTIL A=0
62BA F1                     163         POP  AF
62BB C9                     164         RET
62BC                        165 :
62BC                        166 [INCｴﾗｰ回数]
62BC 3A 16 30 B7 28 04      167         IF (BAT対応)<>0 THEN
62C2 AF CD 33 20            168           A=0  CALL #ERROR  :for BAT
62C6                        169         FI
62C6                        170 :
62C6 3A 08 64 3C 32 08 64   171         INC (ｴﾗｰ回数)
62CD                        172 :       A=(ｴﾗｰ回数)
62CD FE 0A D8               173         IF A<10 RET
62D0                        174 :
62D0 CD C4 1F               175         CALL #BELL
62D3 CD 1C 63               176         CALL @MPRNT
62D6 E6 F3 E0 53 20 21 21   177         DM @TOO,@MANY,@ERR,"S !!",$0D,0
62DD 0D 00
62DF C3 7E 30               178         JP [HOT]
62E2                        179 :
62E2                        180 :
62E2                        181 [ERROR処理]
62E2 E1 22 18 63            182         POP HL  (ERRORﾘﾀｰﾝ)=HL
62E6                        183 :
62E6 CD BC 62               184         [INCｴﾗｰ回数]
62E9                        185 :
62E9 CD C4 1F               186         CALL #BELL
62EC                        187 :
62EC 3A 06 5F FE 00 CC E5   188         IF (ﾘｽﾄ表示SW)=OFF [行表示]
62F3 5E
62F4                        189 :
62F4 DD E5 E1 11 A5 A1 19   190         HL=IX  DE=-LINEBUFF+10  ADD HL,DE
62FB                        191 :       HL=(IX-LINEBUFF)+10
62FB CD F1 1F 28 7C B5 20   192         DO HL {  CALL #PRNTS  }
6302 F8
6303 3E 5E                  193         A="^"
6305 CD F4 1F               194         CALL #PRINT
6308 CD EB 1F               195         CALL #NL
630B                        196 :
630B D1                     197         POP DE
630C                        198         {
630C 1A 13 B7 28 05         199           A=(DE) INC DE  IF A=0 EXIT
6311                        200 :
6311 CD 29 63               201           CALL @PRINT
6314 18 F6                  202         }
6316 D5                     203         PUSH DE
6318                        204 ERRORﾘﾀｰﾝ EQU $+1
6317 21 00 00 E5 C9         205         HL=$0000  PUSH HL  RET
631C                        206 :
631C                        207 :
631C                        208 @MPRNT
631C D1                     209         POP DE
631D                        210         {
631D 1A 13 B7 28 05         211           A=(DE) INC DE  IF A=0 EXIT
6322                        212 :
6322 CD 29 63               213           CALL @PRINT
6325 18 F6                  214         }
```

```
6327 D5                     215         PUSH DE
6328 C9                     216         RET
6329                        217 :
6329                        218 :
6329                        219 @PRINT :in A
6329 FE E0 30 05            220         IF A<$E0 THEN
632D CD F4 1F               221           CALL #PRINT
6330 18 14                  222         ELSE
6332 D5                     223           PUSH DE
6333                        224 :
6333 D6 E0 11 47 63         225           SUB A,$E0  DE=ERRMSGTBL
6338 28 08                  226           IF NZ THEN
633A 47                     227             DO B,A {
633B 1A 13 B7 20 FB         228               { A=(DE) INC DE } UNTIL A=0
6340 10 F9                  229             }
6342                        230           FI
6342 CD E5 1F               231           CALL #MSX
6345                        232 :
6345 D1                     233           POP DE
6346                        234         FI
6346 C9                     235         RET
6347                        236 :
00E0                        237 @ERR        EQU $E0
00E1                        238 @ILL        EQU $E1
00E2                        239 @ARG        EQU $E2
00E3                        240 @OVF        EQU $E3
00E4                        241 @MIS        EQU $E4
00E5                        242 @CNT        EQU $E5
00E6                        243 @TOO        EQU $E6
00E7                        244 @MMATCH     EQU $E7
00E8                        245 @OF         EQU $E8
00E9                        246 @TYP        EQU $E9
00EA                        247 @LNG        EQU $EA
00EB                        248 @STR        EQU $EB
00EC                        249 @UDEF       EQU $EC
00ED                        250 @TBL        EQU $ED
00EE                        251 @NME        EQU $EE
00EF                        252 @CONST      EQU $EF
00F0                        253 @FUNC       EQU $F0
00F1                        254 @SYNTAX     EQU $F1
00F2                        255 @NEST       EQU $F2
00F3                        256 @MANY       EQU $F3
00F4                        257 @STACK      EQU $F4
00F5                        258 @INCL       EQU $F5
00F6                        259 @GLBL       EQU $F6
00F7                        260 @USE        EQU $F7
00F8                        261 @LOCAL      EQU $F8
00F9                        262 @HASH       EQU $F9
00FA                        263 @COMPLETED  EQU $FA
00FB                        264 @REAL       EQU $FB
00FC                        265 @REDEF      EQU $FC
6347                        266 :
6347                        267 ERRMSGTBL
6347 45 52 52 4F 52 00      268         DM "ERROR",0
634D 49 4C 4C 45 47 41 4C   269         DM "ILLEGAL ",0
6354 20 00
6356 41 52 47 00            270         DM "ARG",0
635A 4F 2E 46 2E 00         271         DM "O.F.",0
635F 4D 49 53 53 49 4E 47   272         DM "MISSING ",0
6366 20 00
6368 43 41 4E 27 54 20 00   273         DM "CAN'T ",0
636F 54 4F 4F 20 00         274         DM "TOO ",0
6374 4D 49 53 4D 41 54 43   275         DM "MISMATCHED ",0
637B 48 45 44 20 00
6380 20 4F 46 20 00         276         DM " OF ",0
6385 54 59 50 45 00         277         DM "TYPE",0
638A 4C 4F 4E 47 20 00      278         DM "LONG ",0
6390 53 54 52 49 4E 47 00   279         DM "STRING",0
6397 55 4E 44 45 46 20 00   280         DM "UNDEF ",0
639E 20 54 42 4C 20 00      281         DM " TBL ",0
63A4 4E 41 4D 45 00         282         DM "NAME",0
63A9 43 4F 4E 53 54 20 00   283         DM "CONST ",0
63B0 46 55 4E 43 20 00      284         DM "FUNC ",0
63B6 53 59 4E 54 41 58 20   285         DM "SYNTAX ",0
63BD 00
63BE 4E 45 53 54 49 4E 47   286         DM "NESTING ",0
63C5 20 00
63C7 4D 41 4E 59 20 00      287         DM "MANY ",0
63CD 53 54 41 43 4B 20 00   288         DM "STACK ",0
63D4 49 4E 43 4C 55 44 45   289         DM "INCLUDE ",0
63DB 20 00
63DD 47 4C 4F 42 41 4C 00   290         DM "GLOBAL",0
63E4 55 53 45 00            291         DM "USE",0
63E8 4C 4F 43 41 4C 00      292         DM "LOCAL",0
63EE 48 41 53 48 00         293         DM "HASH",0
63F3 43 4F 4D 50 4C 45 54   294         DM "COMPLETED",0
63FA 45 44 00
63FD 52 45 41 4C 00         295         DM "REAL",0
6402 52 45 44 45 46 00      296         DM "REDEF",0
6408                        297 :
6408 00                     298 ｴﾗｰ回数 DB 0
6409                        299 :
6409                        300 :=====================================
6409                        301 :
6409                        302 [組込登録]
6409 DD E5                  303         PUSH IX
640B                        304 :
640B DD 21 4D 66 01 00 00   305         IX=組込TBL1 BC=0          [組込登録SUB]
6412 CD A3 64
6415 DD 21 6B 66 ED 4B C3   306         IX=組込TBL2 BC=(OBJ先頭) [組込登録SUB]
641C 33 CD A3 64
6420                        307 :ﾗﾝﾀｲﾑﾙｰﾁﾝ内関数登録
6420 DD 21 26 67 CD 44 64   308         IX=組込TBL3           [関数組込登録SUB]
6427                        309 :
6427 2A 59 5A 22 08 5B      310         (関数表TOP)=HL=(大域表ﾎﾟｲﾝﾀ)
642D                        311 :main()登録
642D DD 21 43 66 CD 44 64   312         IX=組込TBL0 [関数組込登録SUB]
6434 21 26 00               313         HL=&MAIN-&R
6437 ED 5B C3 33            314         DE=(OBJ先頭)
643B 19 3E 0A CD 0D 5A      315         ADD HL,DE  A=未関数$ [表変更]
6441                        316 :
6441 DD E1                  317         POP  IX
6443 C9                     318         RET
6444                        319 :
6444                        320 [関数組込登録SUB] :in IX
6444 3E 00 32 51 5A         321         (表域)=大域$
6449                        322         {
6449 DD 7E 00 DD 23 B7 C8   323           A=(IX) INC IX  IF A=0 RET
6450                        324 :
6450 32 52 5A               325           (登録型)=A
6453 CD 1B 38 DD 23         326           [関数名登録] INC IX :for $0D
6458                        327 :
6458 DD 6E 00 DD 23         328           L=(IX) INC IX
645D DD 66 00 DD 23         329           H=(IX) INC IX
6462 ED 4B C3 33            330           BC=(OBJ先頭)
6466 09 3E 09 CD 0D 5A      331           ADD HL,BC  A=関数$ [表変更]
646C                        332 :
646C 2A 18 38 7D D6 0B 6F   333           HL=(関数WKﾎﾟｲﾝﾀ) SUB HL,引数最大数+3
6473 7C DE 00 67
6477 11 B6 04 ED 4B C3 33   334           DE=&TEMP-&R  BC=(OBJ先頭)
647E EB 09 EB               335           ADD DE,BC
6481 DD 46 00 DD 23         336           B=(IX) INC IX
6486 7B CD 5B 45            337           A=E #POKE+ :ｱﾄﾞﾚｽ
648A 7A CD 5B 45            338           A=D #POKE+ :ｱﾄﾞﾚｽ
648E 78 CD 5B 45            339           A=B #POKE+ :数
6492 04 05 28 0B            340           UNTIL B=0 {
6496 DD 7E 00 CD 5B 45 DD   341             A=(IX) #POKE+  INC IX
649D 23
649E 05                     342             DEC B
649F 18 F1                  343           }
64A1 18 A6                  344         }
64A3                        345 :
64A3                        346 [組込登録SUB] :in IX,BC:dataｵﾌｾｯﾄ
64A3 3E 00 32 51 5A         347         (表域)=大域$
64A8                        348         {
64A8 DD 7E 00 DD 23 B7 C8   349           A=(IX) INC IX  IF A=0 RET
64AF 32 52 5A               350           (登録型)=A
64B2                        351 :
64B2 DD 7E 00 DD 23         352           A=(IX) INC IX
64B7 F5                     353           PUSH AF
64B8 11 00 00               354           DE=0
64BB FE 0B 28 08 FE 06 28   355           IF A=M関$ OR A=配列2$ OR A=関変2$ THEN
64C2 04 FE 04 20 05
64C7 DD 5E 00 DD 23         356             E=(IX) INC IX
64CC                        357           FI
64CC DD 6E 00 DD 23         358           L=(IX)  INC IX
64D1 DD 66 00 DD 23 09      359           H=(IX)  INC IX  ADD HL,BC
64D7 F1                     360           POP AF
64D8                        361 :         (表域) (登録型)
64D8                        362 :         A :type
64D8                        363 :         HL:data/ｱﾄﾞﾚｽ
64D8                        364 :         DE:補助data
64D8                        365 :           (M関$引数数/２次元配列第２添字)
64D8 C5                     366           PUSH BC
64D9 CD 94 58 DD 23         367           [表登録]  INC IX :for $0D
64DE C1                     368           POP  BC
64DF 18 C7                  369         }
64E1                        370 :
64E1                        371 :
64E1                        372 [ﾗﾝﾀｲﾑSRB設定]
64E1                        373 :       HL=(OFFSET) DE=(OBJ先頭)
64E1                        374 :       PUSH HL/DE/IY
64E1                        375 :
64E1 FD 21 8A 69 FD 22 C3   376         (OBJ先頭)=IY=&RUNTIME  (OFFSET)=HL=0
64E8 33 21 00 00 22 C5 33
```



▶「エビ天」に第２回アマチュアCGAコンテスト入賞の「超強力宇宙人」が出ていたがウケはあまりよくなかった。まあ，審査員の見るところも違うだろうけど，やっぱりあのオチはよくないよな。
金子　隆博(19)群馬県

```
64EF                     377 :
64EF CD F3 64            378         [ﾗﾝﾀｲﾑ転送]
64F2                     379 :
64F2                     380 :       POP IY/DE/HL
64F2                     381 :       (OFFSET)=HL (OBJ先頭)=DE
64F2 C9                  382         RET
64F3                     383 :
64F3                     384 :
64F3                     385 [ﾗﾝﾀｲﾑ転送]
64F3 21 8A 69            386         HL=&RUNTIME
64F6                     387         {
64F6 CD FC 65 D4 07 65   388           [DATA領域?] IF NC [BYTE]
64FC                     389 :
64FC ED 4B 08 30         390           BC=(ﾗﾝﾀｲﾑ最終ADR)
6500 79 95 78 9C 30 F0   391         } UNTIL BC<HL
6506 C9                  392         RET
6507                     393 :
6507                     394 :
6507                     395 [BYTE]
6507 7E                  396         A=(HL)
6508 FE 40 38 07 FE C0 30 397        IF A>=$40 AND A<$C0 THEN [1byte] RET
650F 03 C3 C2 65
6513                     398 :
6513 FE DD CA 4D 65      399         IF A=$DD [DD&FD命令] RET
6518 FE FD CA 4D 65      400         IF A=$FD [DD&FD命令] RET
651D FE ED CA 3E 65      401         IF A=$ED [ED命令]    RET
6522                     402 :
6522 E6 07 FE 06 CA BF 65 403        AND A,$07  IF A=6 [2byte] RET
6529                     404 :
6529 11 D0 65 CD C7 65 DA 405        DE=DATA3B [SEARCH] IF C [3nn]   RET
6530 6E 65
6532 11 EB 65 CD C7 65 DA 406        DE=DATA2B [SEARCH] IF C [2byte] RET
6539 BF 65
653B                     407 :
653B C3 C2 65            408         [1byte] RET
653E                     409 :
653E                     410 :
653E                     411 [ED命令]
653E CD C2 65            412         [1byte]
6541                     413 :
6541 11 F5 65 CD C7 65 DA 414        DE=DATAED [SEARCH] IF C [3nn]   RET
6548 6E 65
654A                     415 :
654A C3 C2 65            416         [1byte] RET
654D                     417 :
654D                     418 :
654D                     419 [DD&FD命令]
654D CD C2 65            420         [1byte]
6550                     421 :
6550 7E                  422         A=(HL)
6551 FE 30 DA 07 65      423         IF A<$30 [BYTE]  RET
6556 FE 38 DA 68 65      424         IF A<$38 [BYTEd] RET
655B FE 40 DA 07 65      425         IF A<$40 [BYTE]  RET
6560 FE E0 DA 68 65      426         IF A<$E0 [BYTEd] RET
6565 C3 07 65            427                  [BYTE]  RET
6568                     428 :
6568                     429 [BYTEd]
6568 CD 07 65 C3 C2 65   430         [BYTE] [1byte] RET
656E                     431 :
656E                     432 :
656E                     433 [3nn]
656E CD C2 65            434         [1byte]
6571                     435 :
6571 5E 23               436         E=(HL) INC HL
6573 56 23               437         D=(HL) INC HL
6575 ED 4B 08 30         438         BC=(ﾗﾝﾀｲﾑ最終ADR)
6579                     439 :RUNTIME内ﾙｰﾁﾝ
6579 7B D6 8A 7A DE 69 38 440        IF DE>=&RUNTIME AND DE<=BC THEN
6580 17 79 93 78 9A 38 11
6587 ED 4B C3 33         441           BC=(OBJ先頭)
658B 7B D6 8A 5F 7A DE 69 442          SUB DE,&RUNTIME
6592 57
6593 EB 09 EB            443           ADD DE,BC
6596                     444 :SOROBAN設定
6596 18 1F 7B D6 00 7A DE 445        EF DE>=&SRB AND DE<=&SRB+$FF THEN
659D F0 38 17 7B D6 00 7A
65A4 DE F1 30 0F
65A8 ED 4B CC 33         446           BC=(SOROBANｱﾄﾞﾚｽ)
65AC 7B D6 00 5F 7A DE F0 447          SUB DE,&SRB
65B3 57
65B4 EB 09 EB            448           ADD DE,BC
65B7                     449         FI
65B7 7B CD 7B 5B         450         A=E @[IY]A
65BB 7A C3 7B 5B         451         A=D @[IY]A RET
65BF                     452 :
65BF                     453 :
65BF                     454 [2byte]
65BF CD C2 65            455         [1byte]
65C2                     456 :       [1byte]
65C2                     457 :       RET
65C2                     458 :
65C2                     459 [1byte]
65C2 7E 23 C3 7B 5B      460         A=(HL) INC HL  @[IY]A RET
65C7                     461 :
65C7                     462 :
65C7                     463 [SEARCH]
65C7                     464         {
65C7 1A B7 C8            465           A=(DE) IF A=0 RET ;RCF
65CA 13                  466           INC DE
65CB BE 20 F9            467         } UNTIL A=(HL)
65CE                     468 :
65CE 37 C9               469         SCF RET
65D0                     470 :
65D0                     471 DATA3B
65D0 01 11 21 22 2A 31   472         DB $01,$11,$21,$22,$2A,$31
65D6 32 3A C2 C3 C4 CA   473         DB $32,$3A,$C2,$C3,$C4,$CA
65DC CC CD D2 D4 DA DC   474         DB $CC,$CD,$D2,$D4,$DA,$DC
65E2 E2 E4 EA EC F2 F4   475         DB $E2,$E4,$EA,$EC,$F2,$F4
65E8 FA FC               476         DB $FA,$FC
65EA 00                  477         DB 0
65EB                     478 :
65EB                     479 DATA2B
65EB                     480 :       DB $06,$0E,$16,$1E,$26,$2E,$36,$3E
65EB                     481 :       DB $C6,$CE,$D6,$DE,$E6,$EE,$F6,$FE
65EB 10 18 20 28 30 38   482         DB $10,$18,$20,$28,$30,$38
65F1 CB D3 DB            483         DB $CB,$D3,$DB
65F4 00                  484         DB 0
65F5                     485 :
65F5                     486 DATAED
65F5 43 4B 53 5B 73 7B   487         DB $43,$4B,$53,$5B,$73,$7B
65FB 00                  488         DB 0
65FC                     489 :
65FC                     490 :
65FC                     491 [DATA領域?]
65FC 11 29 66            492         DE=DATA領域ﾃﾞｰﾀ
65FF                     493         {
65FF 1A 4F 13            494           C=(DE) INC DE
6602 1A 47 13 78 B1 C8   495           B=(DE) INC DE  IF BC=0 RET :RCF
6608                     496 :
6608 7C B8 20 02 7D B9 28 497          IF HL=BC EXIT
660F 04
6610                     498 :
6610 13 13               499           INC DE/DE
6612 18 EB               500         }
6614 1A 4F 13            501         C=(DE) INC DE
6617 1A 47               502         B=(DE)
6619 03                  503         INC BC
661A 7E 23 CD 7B 5B 7C B8 504        { A=(HL) INC HL @[IY]A } UNTIL HL=BC
6621 20 02 7D B9 20 F3
6627                     505 :
6627 37 C9               506         SCF RET
6629                     507 :
6629                     508 :************************************
6629                     509 :
F000                     510 &SRB  EQU $F000
0001                     511 VS    EQU VOIDS
0002                     512 CS    EQU CHRS
0003                     513 IS    EQU INTS
0004                     514 RS    EQU REALS
6629                     515 :
6629                     516 :
6629                     517 DATA領域ﾃﾞｰﾀ
6629                     518 :       from        to
6629 3B 6E 82 6E         519         DW &WORK1   DW &WORK1END-1
662D 00 00               520         DW 0
662F                     521 :
662F                     522 :拡張用
662F                     523 :
662F 00 00 00 00 00 00 00 524        DS 20 ;5組
6636 00 00 00 00 00 00 00
663D 00 00 00 00 00 00
6643                     525 :
6643                     526 :
6643                     527 :
6643                     528 組込TBL0
6643                     529 :
6643                     530 :main()を組み込む。
6643                     531 :
6643 03 4D 41 49 4E 0D 00 532        DM IS,"MAIN",$D  DW 0  DM 0
664A 00 00
664C 00                  533         DM 0
664D                     534 :
664D                     535 組込TBL1
664D                     536 :
664D                     537 :ランタイムルーチンのアドレスと
664D                     538 :関係のないものを組み込む。
664D                     539 :
664D 03 01 00 00 46 41 4C 540        DM IS,記定$   DW 0     DM "FALSE",$D
6654 53 45 0D
6657 03 01 01 00 54 52 55 541        DM IS,記定$   DW 1     DM "TRUE",$D
665E 45 0D
6660 03 08 00 C4 1F 42 45 542        DM IS,M関$,0 DW #BELL DM "BEEP",$D
6667 45 50 0D
666A 00                  543         DM 0
666B                     544 :
666B                     545 組込TBL2
666B                     546 :
666B                     547 :ランタイムルーチン内にあり、
666B                     548 :引数を持つ関数以外を組み込む。
666B                     549 :
666B 03 08 00 41 03 49 4E 550        DM IS,M関$,0 DW &[INPUT]-&R  DM "INPUT",$D
6672 50 55 54 0D
6676 04 08 00 9B 03 49 4E 551        DM RS,M関$,0 DW &[INPUTR]-&R DM "INPUTR",$D
667D 50 55 54 52 0D
6682 01 08 00 35 00 53 54 552        DM VS,M関$,0 DW &[STOP]-&R   DM "STOP",$D
6689 4F 50 0D
668C 01 08 00 ED 03 47 45 553        DM VS,M関$,0 DW &[GETREG]-&R DM "GETREG",$D
6693 54 52 45 47 0D
6698 03 02 D6 03 5E 42 43 554        DM IS,変数$  DW &[^BC]-&R    DM "^BC",$D
669F 0D
66A0 03 02 D9 03 5E 44 45 555        DM IS,変数$  DW &[^DE]-&R    DM "^DE",$D
66A7 0D
66A8 03 02 DC 03 5E 48 4C 556        DM IS,変数$  DW &[^HL]-&R    DM "^HL",$D
66AF 0D
66B0 03 02 E0 03 5E 49 58 557        DM IS,変数$  DW &[^IX]-&R    DM "^IX",$D
66B7 0D
66B8 03 02 E4 03 5E 49 59 558        DM IS,変数$  DW &[^IY]-&R    DM "^IY",$D
66BF 0D
66C0 03 02 F1 04 5E 41 0D 559        DM IS,変数$  DW &[^A]-&R     DM "^A",$D
66C7 03 02 F0 04 5E 41 46 560        DM IS,変数$  DW &[^AF]-&R    DM "^AF",$D
66CE 0D
66CF 03 02 F3 04 5E 53 50 561        DM IS,変数$  DW &[^SP]-&R    DM "^SP",$D
66D6 0D
66D7 03 02 F5 04 5E 43 59 562        DM IS,変数$  DW &[^CY]-&R    DM "^CY",$D
66DE 0D
66DF 03 02 F7 04 5E 5A 45 563        DM IS,変数$  DW &[^ZERO]-&R  DM "^ZERO",$D
66E6 52 4F 0D
66E9 00                  564         DM 0
66EA                     565 :
66EA                     566 :拡張用
66EA                     567 :
66EA 00 00 00 00 00 00 00 568        DS 60  ;約6個
66F1 00 00 00 00 00 00 00
66F8 00 00 00 00 00 00 00
66FF 00 00 00 00 00 00 00
6706 00 00 00 00 00 00 00
670D 00 00 00 00 00 00 00
6714 00 00 00 00 00 00 00
671B 00 00 00 00 00 00 00
6722 00 00 00 00
6726                     569 :
6726                     570 組込TBL3
6726                     571 :
6726                     572 :ランタイムルーチン内にあり、
6726                     573 :引数を持つ関数を組み込む。
6726                     574 :
6726 03 53 45 58 0D 50 02 575        DM IS,"SEX",$D    DW &[SEX]-&R    DM 1,IS
672D 01 03
672F 03 52 4E 44 0D AF 03 576        DM IS,"RND",$D    DW &[乱数]-&R   DM 1,IS
6736 01 03
6738 03 42 49 54 0D 5B 02 577        DM IS,"BIT",$D    DW &[BIT]-&R    DM 2,IS,IS
673F 02 03 03
6742 01 53 45 54 0D 6D 02 578        DM VS,"SET",$D    DW &[SET]-&R    DM 2,IS,IS
6749 02 03 03
674C 01 52 45 53 45 54 0D 579        DM VS,"RESET",$D  DW &[RESET]-&R  DM 2,IS,IS
6753 87 02 02 03 03
6758 01 57 49 44 54 48 0D 580        DM VS,"WIDTH",$D  DW &[WIDTH]-&R  DM 1,IS
675F A3 02 01 03
6763 01 4C 4F 43 41 54 45 581        DM VS,"LOCATE",$D DW &[LOCATE]-&R DM 2,IS,IS
676A 0D AA 02 02 03 03
6770 03 53 43 52 45 45 4E 582        DM IS,"SCREEN",$D DW &[SCREEN]-&R DM 2,IS,IS
6777 0D B5 02 02 03 03
677D 01 50 52 4D 4F 44 45 583        DM VS,"PRMODE",$D DW &[PRMODE]-&R DM 1,IS
6784 0D C4 02 01 03
6789 03 49 4E 4B 45 59 0D 584        DM IS,"INKEY",$D  DW &[INKEY]-&R  DM 1,IS
6790 E1 02 01 03
6794 03 47 45 54 4C 0D 01 585        DM IS,"GETL",$D   DW &[GETL]-&R   DM 1,IS
679B 03 01 03
679E 03 47 45 54 4C 49 4E 586        DM IS,"GETLIN",$D DW &[GETLIN]-&R DM 2,IS,IS
67A5 0D 06 03 02 03 03
67AB 03 4C 49 4E 50 55 54 587        DM IS,"LINPUT",$D DW &[LINPUT]-&R DM 2,IS,IS
67B2 0D FB 02 02 03 03
67B8 03 43 41 4C 4C 0D CD 588        DM IS,"CALL",$D   DW &[CALL]-&R   DM 1,IS
67BF 03 01 03
67C2                     589 :
67C2 03 43 56 49 54 53 0D 590        DM IS,"CVITS",$D  DW &[CVITS]-&R  DM 2,IS,IS
67C9 22 02 02 03 03
67CE 03 43 56 52 54 53 0D 591        DM IS,"CVRTS",$D  DW &[CVRTS]-&R  DM 2,RS,IS
67D5 14 02 02 04 03
67DA 04 43 56 53 54 52 0D 592        DM RS,"CVSTR",$D  DW &[CVSTR]-&R  DM 1,IS
67E1 A0 03 01 03
67E5                     593 :
67E5 04 46 49 58 0D 51 04 594        DM RS,"FIX",$D    DW &[FIX]-&R    DM 1,RS
67EC 01 04
67EE 04 46 52 41 43 0D 63 595        DM RS,"FRAC",$D   DW &[FRAC]-&R   DM 1,RS
67F5 04 01 04
67F8 04 49 4E 54 0D 57 04 596        DM RS,"INT",$D    DW &[INT]-&R    DM 1,RS
67FF 01 04
6801 04 43 49 4E 54 0D 5D 597        DM RS,"CINT",$D   DW &[CINT]-&R   DM 1,RS
6808 04 01 04
680B 04 53 51 52 0D 45 04 598        DM RS,"SQR",$D    DW &[SQR]-&R    DM 1,RS
6812 01 04
6814 04 53 49 4E 0D 21 04 599        DM RS,"SIN",$D    DW &[SIN]-&R    DM 1,RS
681B 01 04
681D 04 43 4F 53 0D 27 04 600        DM RS,"COS",$D    DW &[COS]-&R    DM 1,RS
6824 01 04
6826 04 54 41 4E 0D 2D 04 601        DM RS,"TAN",$D    DW &[TAN]-&R    DM 1,RS
682D 01 04
682F 04 41 54 4E 0D 33 04 602        DM RS,"ATN",$D    DW &[ATN]-&R    DM 1,RS
6836 01 04
6838 04 45 58 50 0D 39 04 603        DM RS,"EXP",$D    DW &[EXP]-&R    DM 1,RS
683F 01 04
6841 04 4C 4F 47 0D 3F 04 604        DM RS,"LOG",$D    DW &[LOG]-&R    DM 1,RS
6848 01 04
684A 04 50 4F 57 0D 69 04 605        DM RS,"POW",$D    DW &[POW]-&R    DM 2,RS,RS
6851 02 04 04
6854 04 52 41 44 0D 4B 04 606        DM RS,"RAD",$D    DW &[RAD]-&R    DM 1,RS
685B 01 04
685D 00                  607         DM 0
685E                     608 :
685E                     609 :拡張用
685E                     610 :
685E 00 00 00 00 00 00 00 611        DS 300-18
6865 00 00 00 00 00 00 00
686C 00 00 00 00 00 00 00
6873 00 00 00 00 00 00 00
687A 00 00 00 00 00 00 00
6881 00 00 00 00 00 00 00
6888 00 00 00 00 00 00 00
688F 00 00 00 00 00 00 00
6896 00 00 00 00 00 00 00
689D 00 00 00 00 00 00 00
68A4 00 00 00 00 00 00 00
68AB 00 00 00 00 00 00 00
68B2 00 00 00 00 00 00 00
68B9 00 00 00 00 00 00 00
68C0 00 00 00 00 00 00 00
68C7 00 00 00 00 00 00 00
68CE 00 00 00 00 00 00 00
68D5 00 00 00 00 00 00 00
68DC 00 00 00 00 00 00 00
68E3 00 00 00 00 00 00 00
68EA 00 00 00 00 00 00 00
68F1 00 00 00 00 00 00 00
68F8 00 00 00 00 00 00 00
68FF 00 00 00 00 00 00 00
6906 00 00 00 00 00 00 00
690D 00 00 00 00 00 00 00
6914 00 00 00 00 00 00 00
691B 00 00 00 00 00 00 00
6922 00 00 00 00 00 00 00
6929 00 00 00 00 00 00 00
6930 00 00 00 00 00 00 00
6937 00 00 00 00 00 00 00
693E 00 00 00 00 00 00 00
6945 00 00 00 00 00 00 00
694C 00 00 00 00 00 00 00
6953 00 00 00 00 00 00 00
695A 00 00 00 00 00 00 00
6961 00 00 00 00 00 00 00
6968 00 00 00 00 00 00 00
696F 00 00 00 00 00 00 00
6976 00 00
6978                     612 :
6978                     613 :
6978                     614 @PATCH2
6978 ED 5B C5 33 19      615         DE=(OFFSET) ADD HL,DE
697D 0B 5E 0A C9         616         DEC BC  E=(HL)  A=(BC)  RET
6981                     617 :
6981                     618 @PATCH1
6981 ED 5B C5 33 19      619         DE=(OFFSET) ADD HL,DE
6985 5E 71 23 C9         620         E=(HL)  (HL)=C  INC HL  RET
698A                     621 :
698A                     622 :毎度おなじみBUGパッチ用エリアです。
698A                     623 :ここから上に向かって伸びていきます。
698A                     624 :
698A                     625 :まさか掲載時に使うはめになろうとは…
698A                     626 :
698A                     627 :************************************
698A                     628 :
698A                     629 :       ランタイム・ルーチン
698A                     630 :
698A                     631 &RUNTIME
698A                     632 &R
698A D9                  633         EXX
698B E1 22 BD 69         634         POP HL (&RET_ADR)=HL
698F D9                  635         EXX
6990                     636 :
6990 C5 D5 DD E5 FD E5   637         PUSH BC/DE/IX/IY  :HLは保存しない
6996 ED 73 B4 69         638         (&SP)=SP
699A                     639 :
699A                     640 &STACK
699A                     641 :       SP=$0000
699A 00 00 00            642         DB 0,0,0
699D                     643 :
699D CD C2 69 28 1D      644         &[SRBCHECK]   IF Z     JR &[STOP]
69A2 21 00 F0 7A BC 20 02 645        HL=#PRCSN-&SRB IF DE<>HL JR &[STOP]
69A9 7B BD 20 12
69AD                     646 :
69AD 36 08               647         (HL)=8
69AE                     648 精度     EQU $-1
69AF                     649 :       CALL &MAIN()
69AF CD 00 00            650         CALL $0000
69B0                     651 &MAIN    EQU $-2
69B2 B7                  652         RCF
69B3                     653 :
69B3                     654 &RET
69B3 31 00 00            655         SP=$0000
69B4                     656 &SP      EQU $-2
69B6 FD E1 DD E1 D1 C1   657         POP IY/IX/DE/BC
69BC                     658 :
69BC C3 00 00            659         JP $0000
69BD                     660 &RET_ADR EQU $-2
69BF                     661 :
69BF                     662 &[STOP]
69BF 37 18 F1            663         SCF JR &RET
69C2                     664 :
69C2                     665 &[SRBCHECK] :out Z,DE
69C2 2A 82 1F 7C B5 C8   666         HL=($1F82) IF HL=0 RET
69C8 23 23               667         INC HL/HL
69CA 5E 23               668         E=(HL) INC HL
69CC 56                  669         D=(HL)
69CD 7A B3               670         A=D OR E
69CF C9                  671         RET
69D0                     672 :
69D0                     673 :
69D0                     674 &[SOS参照]
69D0 CD 94 1F 5F 16 00 EB 675        CALL #PEEK  E=A  D=0    EX DE,HL RET
69D7 C9
69D8                     676 &[SOSW参照]
69D8 CD 94 1F 5F 23      677         CALL #PEEK  E=A  INC HL
69DD CD 94 1F 57 2B EB C9 678        CALL #PEEK  D=A  DEC HL  EX DE,HL RET
69E4                     679 :
69E4                     680 &[SOSW代入]
69E4 7B CD 9A 1F 23      681         A=E  CALL #POKE  INC HL
69E9 7A 18 01            682         A=D  JR &[SOS代入]1
69EC                     683 &[SOS代入]
69EC 7B                  684         A=E
69ED                     685 &[SOS代入]1
69ED CD 9A 1F EB C9      686         CALL #POKE  EX DE,HL  RET
69F2                     687 :
69F2                     688 &[PORT参照]
69F2 44 4D ED 68 26 00 50 689        BC=HL  IN L,(BC) H=0  DE=BC  RET
69F9 59 C9
69FB                     690 &[PORTW参照]
69FB 44 4D               691         BC=HL
69FD ED 68 03            692         IN L,(BC) INC BC
6A00 ED 60 0B 50 59 C9   693         IN H,(BC) DEC BC  DE=BC  RET
6A06                     694 :
6A06                     695 &[PORT代入]
6A06 44 4D ED 59 EB C9   696         BC=HL  OUT (BC),E  EX DE,HL  RET
6A0C                     697 &[PORTW代入]
6A0C 44 4D ED 59 03 ED 51 698        BC=HL  OUT (BC),E  INC BC  OUT (BC),D
6A13 EB C9               699         EX DE,HL  RET
6A15                     700 :
6A15                     701 :
6A15                     702 &MOVE_R
6A15 EB                  703         EX DE,HL
6A16                     704 &MOVE_EX
6A16 CD 02 F0 EB C9      705         CALL #MOVE-&SRB  EX DE,HL  RET
6A1B                     706 :
6A1B                     707 &CVUTF
6A1B EB C3 11 F0         708         EX DE,HL  CALL #CVUTF-&SRB  RET
6A1F                     709 :
6A1F                     710 &CVITF
6A1F EB C3 14 F0         711         EX DE,HL  CALL #CVITF-&SRB  RET
6A23                     712 :
6A23                     713 :
6A23                     714 &[乗算] ; HL*DE => HL
6A23 7D 93 7C 9A 30 01 EB 715        IF HL<DE THEN  EX DE,HL
6A2A 44 4D 21 00 00      716         BC=HL  HL=0
6A2F 7A                  717         A=D
6A30 B7 28 0D            718         IF A<>0 THEN
6A33                     719           {
6A33                     720 :           CY=0
6A33 1F CB 1B            721             RRA  RR E       : CY(0)>>A>>E>>CY
6A36 30 01 09            722             IF C THEN ADD HL,BC
6A39 CB 21 CB 10         723             SLA C  RL B     : CY<<B<<C<<0
6A3D B7 20 F3            724           } UNTIL A=0
6A40                     725         FI
6A40 7B                  726         A=E
6A41                     727         {
6A41                     728 :         CY=0
6A41 1F                  729           RRA               : CY(0)>>A>>CY
6A42 30 01 09            730           IF C THEN ADD HL,BC
6A45 CB 21 CB 10         731           SLA C  RL B       : CY<<B<<C<<0
6A49 B7 20 F5            732         } UNTIL A=0
6A4C C9                  733         RET
6A4D                     734 :
6A4D                     735 :
6A4D                     736 &[除算] ; HL/DE -> HL...DE
6A4D                     737 :
6A4D 7D 93 7C 9A 30 05 EB 738        IF HL<DE THEN  EX DE,HL  HL=0  RET
6A54 21 00 00 C9
6A58                     739 :
6A58 14 15 28 18         740         IF D=0 JR &[除算8]
6A5C                     741 :
6A5C 4D                  742         C=L
6A5D 6C                  743         L=H
6A5E AF 67               744         A=0  H=A
6A60 06 08               745         DO B,8 {
6A62 87                  746           ADD A,A
6A63 CB 21 ED 6A         747           SLA C  ADC HL,HL  : H << L << C
6A67 ED 52               748           SBC HL,DE
6A69 30 02 19 3D         749           IF C THEN  ADD HL,DE  DEC A
6A6D 3C                  750           INC A
6A6E 10 F2               751         }
6A70 EB                  752         EX DE,HL  ; DE:余り
6A71 60 6F               753         H=B  L=A
6A73 C9                  754         RET
6A74                     755 :
6A74                     756 :
6A74                     757 &[除算8]  ; HL/E -> HL...E
6A74 1C 1D 20 05 EB 21 00 758        IF E=0 THEN  EX DE,HL  HL=0  RET
6A7B 00 C9
6A7D AF                  759         A=0
6A7E 06 10 24 25 20 04 06 760        B=16  IF H=0 THEN  B=8  H=L  L=A :0
6A85 08 65 6F
6A88                     761         DO B {
6A88 29                  762           ADD HL,HL
6A89 8F                  763           ADC A,A
6A8A BB 38 02 93 2C      764           IF A>=E THEN  SUB A,E  INC L
6A8F 10 F7               765         }
6A91 5F                  766         E=A ;余り
6A92 C9                  767         RET
6A93                     768 :
6A93                     769 :
6A93                     770 &[剰余]
6A93 CD 4D 6A EB C9      771         &[除算] EX DE,HL RET
6A98                     772 :
6A98                     773 :
6A98                     774 &[SGN]
6A98                     775 :0
6A98 7C B5 C8            776         IF HL=0 RET
6A9B                     777 :1
6A9B CB 7C 21 01 00 C8   778         BIT 7,H  HL=1 IF Z RET
6AA1                     779 :-1
6AA1 2B 2B               780         DEC HL/HL
6AA3 C9                  781         RET
6AA4                     782 :
6AA4                     783 &[ABS]
6AA4 CB 7C C8            784         BIT 7,H  IF Z RET
6AA7                     785 :       &[単項-]
6AA7                     786 :       RET]
6AA7                     787 :
```

▶そろそろ就職活動を開始しようと思う。ということで，どこかいいところはありませんか？ なんてことはいいませんよ。がんばっていいとこ行くぞ。　大西　征二(19)石川県

```
6AA7                    788 &[単項-] :2の補数
6AA7 2B                 789         DEC HL
6AA8                    790 &[CPL]   :1の補数
6AA8 7C 2F 67           791         A=H CPL H=A
6AAB 7D 2F 6F           792         A=L CPL L=A
6AAE C9                 793         RET
6AAF                    794 :
6AAF 7D AB 6F 7C AA 67 C9  795 &[ビットXOR] XOR HL,DE  RET
6AB6 7B CB 3C CB 1D 3D 20  796 &[>>]    DO A,E { SRL H  RR L } RET
6ABD F9 C9
6ABF 7B 29 3D 20 FC C9  797 &[<<]    DO A,E { ADD HL,HL   } RET
6AC5                    798 :
6AC5                    799 &実[論理OR]
6AC5 1A B6 18 1F        800         A=(DE) OR (HL)       JR NZ_TRUE
6AC9                    801 &整[論理OR]
6AC9 7C B5 B2 B3 18 19  802         A=H OR L OR D OR E  JR NZ_TRUE
6ACF                    803 &実[論理AND]
6ACF 7E B7 21 00 00 C8  804         A=(HL) OR A  HL=0  IF Z RET
6AD5 1A B7 18 0F        805         A=(DE) OR A          JR NZ_TRUE
6AD9                    806 &整[論理AND]
6AD9 7C B5 C8           807         A=H OR L  IF Z RET
6ADC 7A B3 18 08        808         A=D OR E             JR NZ_TRUE
6AE0                    809 :
6AE0                    810 &実[<>]
6AE0 CD 32 F0 18 03     811         CALL #CMP+&SRB  JR NZ_TRUE
6AE5                    812 :
6AE5                    813 &整[<>]
6AE5 B7 ED 52           814         SUB HL,DE
6AE8                    815 NZ_TRUE
6AE8 21 01 00 C0        816         HL=1  IF NZ RET
6AEC 2B                 817         DEC HL
6AED C9                 818         RET
6AEE                    819 :
6AEE                    820 &実[==]
6AEE CD 32 F0 18 03     821         CALL #CMP+&SRB  JR Z_TRUE
6AF3                    822 :
6AF3                    823 &整[==]
6AF3 B7 ED 52           824         SUB HL,DE
6AF6                    825 Z_TRUE
6AF6 21 01 00 C8        826         HL=1  IF Z RET
6AFA 2B                 827         DEC HL
6AFB C9                 828         RET
6AFC                    829 :
6AFC 7C B5 18 F6        830 &整[NOT]  A=H    OR L  JR Z_TRUE
6B00 7E B7 18 F2        831 &実[NOT]  A=(HL) OR A  JR Z_TRUE
6B04                    832 :
6B04                    833 &実[<=]
6B04 EB                 834         EX DE,HL
6B05                    835 &実[>=]
6B05 CD 32 F0 18 04     836         CALL #CMP+&SRB  JR NC_TRUE
6B0A                    837 :
6B0A                    838 &整[<=]
6B0A EB                 839         EX DE,HL
6B0B                    840 &整[>=]
6B0B B7 ED 52           841         SUB HL,DE
6B0E                    842 NC_TRUE
6B0E 21 01 00 D0        843         HL=1  IF NC RET
6B12 2B                 844         DEC HL
6B13 C9                 845         RET
6B14                    846 :
6B14                    847 &実[>]
6B14 EB                 848         EX DE,HL
6B15                    849 &実[<]
6B15 CD 32 F0 18 04     850         CALL #CMP+&SRB  JR C_TRUE
6B1A                    851 :
6B1A                    852 &整[>]
6B1A EB                 853         EX DE,HL
6B1B                    854 &整[<]
6B1B B7 ED 52           855         SUB HL,DE
6B1E                    856 C_TRUE
6B1E 21 01 00 D8        857         HL=1  IF C RET
6B22 2B                 858         DEC HL
6B23 C9                 859         RET
6B24                    860 :
6B24                    861 ;PRINT関数
6B24                    862 :
6B24                    863 &[?STR]
6B24                    864         {
6B24 7A B3 C8           865           IF DE=0 RET
6B27 7D                 866           A=L
6B28 CD 4D 6B           867           &[PRT]
6B2B 1B                 868           DEC DE
6B2C 18 F6              869         }
6B2E                    870 :
6B2E 7C CD 4D 6B 7D 18 18  871 &[?CHR] A=H &[PRT]  A=L JR &[PRT]
6B35 3E 0D 18 14        872 &[CR]   A=$0D           JR &[PRT]
6B39                    873 :
6B39                    874 &[?HEX4]
6B39 7C CD 3E 6B        875         A=H &[?HEX]
6B3D                    876 &[?HEX2]
6B3D 7D                 877         A=L
6B3E                    878 &[?HEX]
6B3E F5                 879         PUSH AF
6B3F 0F 0F 0F 0F        880         RRCA RRCA RRCA RRCA
6B43 CD BB 1F           881         CALL #ASC
6B46 CD 4D 6B           882         &[PRT]
6B49                    883 :
6B49 F1                 884         POP AF
6B4A CD BB 1F           885         CALL #ASC
6B4D                    886 :       &[PRT]
6B4D                    887 :       RET
6B4D                    888 :
6B4D                    889 &[PRT] ;AF以外保存
6B4D C3 F4 1F           890         JP #PRINT
6B50                    891 :
6B50                    892 :
6B50                    893 &[?符号付]
6B50 CB 7C 28 0B        894         BIT 7,H  IF Z JR &[?10]
6B54                    895 :
6B54 3E 2D CD 4D 6B     896         A="-" &[PRT]
6B59 CD A7 6A           897         &[単項-]
6B5C                    898 :       &[?10]
6B5C                    899 :       RET
6B5C                    900 :
6B5C 11 FF FF 18 03     901 &[?10]    DE=-1 JR &[?10_n]
6B61 11 05 00           902 &[?10_5]  DE=5
6B64                    903 &[?10_n]
6B64 D5                 904         PUSH DE
6B65 11 3A 6E CD B3 6B  905         DE=&WORK10 &[10進化]
6B6B EB                 906         EX DE,HL
6B6C                    907 :       HL=&WORK10
6B6C D1                 908         POP  DE
6B6D 7B                 909         A=E
6B6E FE 05 30 09        910         IF A<5 THEN
6B72 3E 05 93 23 3D 20 FC  911        A=5 SUB E  DO A { INC HL }
6B79 18 0C FE FF 20 08  912         EF A=-1 THEN
6B7F 7E FE 20 20 03 23 18  913        WHILE (HL)=" "  { INC HL }
6B86 F8
6B87                    914         FI
6B87                    915 :       JR &[?MSX]
6B87                    916 :
6B87 06 00 18 02        917 &[?MSX] B=0   JR &[?MSG]1
6B8B 06 0D              918 &[?MSG] B=$0D
6B8D                    919 &[?MSG]1
6B8D                    920         {
6B8D 7E B8 C8           921           A=(HL) IF A=B RET
6B90 CD 4D 6B           922           &[PRT]
6B93 23                 923           INC HL
6B94 18 F7              924         }
6B96                    925 :
6B96                    926 &[?REAL]
6B96 11 58 6E CD A1 6B 18  927       DE=&WORKREAL  &[CVRTS]処理  JR &[?MSX]
6B9D E9
6B9E                    928 :
6B9E                    929 ;INT CVRTS(REAL 値, INT BUFF):
6B9E                    930 :
6B9E                    931 :       (* BUFFは18 or 34バイト必要 *)
6B9E                    932 :
6B9E                    933 &[CVRTS]
6B9E CD F9 6D           934         &[GETARG2]
6BA1                    935 &[CVRTS]処理
6BA1 CD 17 F0           936         CALL #CVFTS+&SRB
6BA4 EB                 937         EX DE,HL
6BA5 7E FE 20 20 01 23  938         IF (HL)=" " THEN INC HL
6BAB C9                 939         RET
6BAC                    940 :
6BAC                    941 ;INT CVITS(INT 値,BUFF):
6BAC                    942 :
6BAC                    943 :       (* BUFFは6バイト必要 *)
6BAC                    944 :
6BAC                    945 &[CVITS]
6BAC 2A 40 6E           946         HL=(&TEMP)
6BAF ED 5B 42 6E        947         DE=(&TEMP+2)
6BB3                    948 &[10進化]
6BB3                    949 :       PUSH DE
6BB3 E5 D9 E1 D9        950         PUSH HL  EXX  POP HL  EXX
6BB7                    951 :
6BB7 21 05 00 19 36 00  952         HL=5  ADD HL,DE  (HL)=0
6BBD 06 05              953         DO B,5 {
6BBF D9                 954           EXX
6BC0 1E 0A CD 74 6A     955             E=10 &[除算8]
6BC5                    956 :           余り
6BC5 7B C6 30           957             A=E ADD "0"
6BC8 D9                 958           EXX
```

```
6BC9 2B 77              959             DEC HL  (HL)=A
6BCB 10 F2              960         }
6BCD 06 04              961         DO B,4 {
6BCF 7E FE 30 20 05     962           IF (HL)<>"0" EXIT
6BD4 36 20              963           (HL)=" "
6BD6 23                 964           INC HL
6BD7 10 F6              965         }
6BD9                    966 :       POP DE
6BD9 C9                 967         RET
6BDA                    968 :
6BDA                    969 :       int SEX(int 値)
6BDA                    970 :
6BDA                    971 &[SEX]
6BDA 3A 40 6E 6F        972         L=(&TEMP)
6BDE CB 7D              973         BIT 7,L
6BE0 26 00 C8           974         H=0 IF Z RET
6BE3 25 C9              975         DEC H    RET
6BE5                    976 :
6BE5                    977 :       int BIT(int 値,n)
6BE5                    978 :
6BE5                    979 &[BIT]
6BE5 2A 40 6E ED 5B 42 6E  980      HL=(&TEMP)  DE=(&TEMP+2)  &[>>]
6BEC CD B6 6A
6BEF CB 45              981         BIT 0,L
6BF1 21 00 00 C8        982         HL=0 IF Z RET
6BF5 23                 983         INC HL
6BF6 C9                 984         RET
6BF7                    985 :
6BF7                    986 :       int SET(int 値,n)
6BF7                    987 :
6BF7                    988 &[SET]
6BF7 ED 5B 40 6E 2A 42 6E  989      DE=(&TEMP)  HL=(&TEMP+2)
6BFE                    990 :
6BFE 7D E6 0F           991         A=L AND $0F
6C01 21 01 00           992         HL=1
6C04 28 04 29 3D 20 FC  993         IF NZ THEN DO A { ADD HL,HL }
6C0A                    994 :       &[ビットOR]
6C0A                    995 :       RET
6C0A                    996 &[ビットOR]
6C0A 7D B3 6F 7C B2 67 C9  997      OR HL,DE  RET
6C11                    998 :
6C11                    999 :       int RESET(int 値,n)
6C11                   1000 :
6C11                   1001 &[RESET]
6C11 ED 5B 40 6E 2A 42 6E 1002      DE=(&TEMP)  HL=(&TEMP+2)
6C18                   1003 :
6C18 7D E6 0F          1004         A=L AND $0F
6C1B 21 FE FF          1005         HL=$FFFF-1
6C1E 28 06 37 ED 6A 3D 20 1006      IF NZ THEN DO A { SCF  ADC HL,HL }
6C25 FA
6C26                   1007 :       &[ビットAND]
6C26                   1008 :       RET
6C26                   1009 &[ビットAND]
6C26 7D A3 6F 7C A2 67 C9 1010      AND HL,DE  RET
6C2D                   1011 :
6C2D                   1012 :       void WIDTH(int n)
6C2D                   1013 :
6C2D                   1014 &[WIDTH]
6C2D 3A 40 6E 6F C3 30 20 1015      L=(&TEMP)  CALL #WIDCH  RET
6C34                   1016 :
6C34                   1017 :       void LOCATE(int x,y)
6C34                   1018 :
6C34                   1019 &[LOCATE]
6C34 3A 40 6E 6F       1020         L=(&TEMP)
6C38 3A 42 6E 67 C3 1E 20 1021      H=(&TEMP+2)  CALL #LOC  RET
6C3F                   1022 :
6C3F                   1023 :       int SCREEN(int x,y)
6C3F                   1024 :
6C3F                   1025 &[SCREEN]
6C3F 3A 40 6E 6F       1026         L=(&TEMP)
6C43 3A 42 6E 67       1027         H=(&TEMP+2)
6C47 CD 18 20 6F 26 00 C9 1028      CALL #SCRN  L=A  H=0  RET
6C4E                   1029 :
6C4E                   1030 :       void PRMODE(int mode)
6C4E                   1031 :
6C4E                   1032 &[PRMODE]
6C4E 3A 40 6E FE 01    1033         A=(&TEMP) CP 1
6C53                   1034 :
6C53 21 F4 1F          1035         HL=#PRINT
6C56 30 05 CD D6 1F    1036         IF < : CALL #LPTOF
6C5B 18 0A 20 05 CD D9 1F 1037      EF = : CALL #LPTON
6C62 18 03 21 DC 1F    1038         ELSE : HL=#LPRNT
6C67                   1039         FI
6C67 22 4E 6B          1040         (&[PRT]+1)=HL
6C6A C9                1041         RET
6C6B                   1042 :
6C6B                   1043 :       int INKEY(int n)
6C6B                   1044 :
6C6B                   1045 &[INKEY]
6C6B 3A 40 6E FE 01    1046         A=(&TEMP) CP 1
6C70                   1047 :
6C70 30 05 CD D0 1F    1048         IF < : CALL #GETKY
6C75 18 0A 20 05 CD 21 20 1049      EF = : CALL #FLGET
6C7C 18 03 CD CA 1F    1050         ELSE : CALL #INKEY
6C81                   1051         FI
6C81                   1052 :
6C81 6F 26 00 C9       1053         L=A  H=0  RET
6C85                   1054 :
6C85                   1055 :       int LINPUT(int アドレス,長さ)
6C85                   1056 :
6C85                   1057 &[LINPUT]
6C85 CD 18 20 55 18 07 1058         CALL #CSR D=L  JR &[GETLIN]1
6C8B                   1059 :
6C8B                   1060 :       int GETL(int アドレス)
6C8B                   1061 :
6C8B                   1062 &[GETL]
6C8B 11 00 00 18 06    1063         DE=0  JR &[GETLIN]2
6C90                   1064 :
6C90                   1065 :       int GETLIN(int アドレス,長さ)
6C90                   1066 :
6C90                   1067 &[GETLIN]
6C90 16 00             1068         D=0
6C92                   1069 &[GETLIN]1
6C92 3A 42 6E 5F       1070         E=(&TEMP+2)
6C96                   1071 &[GETLIN]2
6C96 2A 40 6E          1072         HL=(&TEMP)
6C99                   1073 &[GETLIN]3
6C99 D5                1074         PUSH DE
6C9A ED 5B 76 1F CD D3 1F 1075      DE=(#KBFAD)  CALL #GETL
6CA1 C1                1076         POP  BC
6CA2                   1077 ;break
6CA2 1A                1078         A=(DE)
6CA3 FE 1B 20 05 77 21 FF 1079      IF A=$1B THEN (HL)=A HL=-1 RET
6CAA FF C9
6CAC                   1080 :
6CAC 04 05 28 08       1081         UNTIL B=0 {
6CB0 1A B7 28 04       1082           IF (DE)=0 EXIT
6CB4 13                1083           INC DE
6CB5 05                1084           DEC B
6CB6 18 F4             1085         }
6CB8 06 00             1086         B=0
6CBA                   1087         DO C {
6CBA 1A 13 B7 28 06    1088           A=(DE) INC DE  IF A=0 EXIT
6CBF 77 23             1089           (HL)=A INC HL
6CC1 04                1090           INC B
6CC2 0D 20 F5          1091         }
6CC5 36 00             1092         (HL)=0
6CC7                   1093 :
6CC7 68 26 00 C9       1094         L=B H=0 RET
6CCB                   1095 :
6CCB                   1096 :       int INPUT()
6CCB                   1097 :
6CCB                   1098 &[INPUT]
6CCB CD 07 6D          1099         &[INPUT入力処理]
6CCE                   1100 ;16進数
6CCE 1A FE 24 20 07    1101         IF (DE)="$" THEN
6CD3 13                1102           INC DE
6CD4 CD B2 1F 38 1C    1103           CALL #HLHEX  IF C JR &INPUT_ERR
6CD9 C9                1104           RET
6CDA                   1105         FI
6CDA                   1106 ;10進数
6CDA 21 00 00          1107         HL=0
6CDD 1A CD 00 6D 38 12 1108         A=(DE) &ADECI  IF C JR &INPUT_ERR
6CE3                   1109         {
6CE3 29 44 4D          1110           ADD HL,HL  BC=HL
6CE6 29                1111           ADD HL,HL
6CE7 29                1112           ADD HL,HL
6CE8 09                1113           ADD HL,BC
6CE9 06 00 4F 09       1114           B=0  C=A  ADD HL,BC
6CED                   1115 :
6CED 13                1116           INC DE
6CEE 1A CD 00 6D       1117           A=(DE) &ADECI
6CF2 30 EF             1118         } UNTIL C
6CF4 C9                1119         RET
6CF5                   1120 :
6CF5                   1121 &INPUT_ERR
6CF5 01 01 00 ED 43 7F 6E 1122      BC=1 (&[^CY])=BC  HL=0 RET
6CFC 21 00 00 C9
6D00                   1123 :
6D00                   1124 &ADECI
6D00 D6 30 D8          1125         SUB A,"0"  IF C RET
6D03 FE 0A             1126         CP A,10
6D05 3F                1127         CCF
6D06 C9                1128         RET
```

```
6D07                   1129 :
6D07                   1130 &[INPUT入力処理]
6D07 01 00 00 ED 43 7F 6E 1131      BC=0 (&[^CY])=BC
6D0E                   1132 :
6D0E CD 18 20          1133         CALL #CSR
6D11 55 1E 00 2A 76 1F CD 1134      D=L E=0  HL=(#KBFAD) &[GETLIN]3
6D18 99 6C
6D1A                   1135 :
6D1A ED 5B 76 1F       1136         DE=(#KBFAD)
6D1E                   1137         {
6D1E 1A FE 20 C0       1138           IF (DE)<>" " RET
6D22 13                1139           INC DE
6D23 18 F9             1140         }
6D25                   1141 :
6D25                   1142 ;REAL INPUTR()
6D25                   1143 :
6D25                   1144 &[INPUTR]
6D25 CD 07 6D 18 04    1145         &[INPUT入力処理] JR &[CVSTR]1
6D2A                   1146 :
6D2A                   1147 ;REAL CVSTR(INT BUFF):
6D2A                   1148 :
6D2A                   1149 &[CVSTR]
6D2A ED 5B 40 6E       1150         DE=(&TEMP)
6D2E                   1151 &[CVSTR]1
6D2E 21 42 6E CD 0E F0 1152         HL=&TEMP+2  CALL #CVSTF+&SRB
6D34 ED 53 63 6D       1153         (&[^DE])=DE
6D38 C9                1154         RET
6D39                   1155 :
6D39                   1156 :       int RND(int n)
6D39                   1157 :
6D39                   1158 &[乱数]
6D39 2A 38 6E 11 83 03 CD 1159      HL=(&旧乱数) DE=899 &[乗算]
6D40 23 6A
6D42                   1160 :
6D42 6C ED 5F 84 67    1161         L=H  A=R ADD H  H=A
6D47                   1162 :
6D47 22 38 6E          1163         (&旧乱数)=HL
6D4A                   1164 :
6D4A ED 5B 40 6E 7A B3 20 1165      DE=(&TEMP)  IF DE=0 THEN EX DE,HL RET
6D51 02 EB C9
6D54                   1166 :
6D54 C3 93 6A          1167         &[剰余] RET
6D57                   1168 :
6D57                   1169 :       int CALL(int アドレス)
6D57                   1170 :
6D57                   1171 &[CALL]
6D57 11 71 6D D5 E5    1172         DE=&[CALL_RET] PUSH DE/HL
6D5C                   1173 :
6D5C 3A 7B 6E          1174         A=(&[^A])
6D5F 01 00 00          1175         BC=$0000
6D60                   1176 &[^BC] EQU $-2
6D62 11 00 00          1177         DE=$0000
6D63                   1178 &[^DE] EQU $-2
6D65 21 00 00          1179         HL=$0000
6D66                   1180 &[^HL] EQU $-2
6D68 DD 21 00 00       1181         IX=$0000
6D6A                   1182 &[^IX] EQU $-2
6D6C FD 21 00 00       1183         IY=$0000
6D6E                   1184 &[^IY] EQU $-2
6D70                   1185 :
6D70 C9                1186         RET
6D71                   1187 &[CALL_RET]
6D71 E5                1188         PUSH HL
6D72                   1189 :         *
6D72 CD 77 6D          1190         &[GETREG]
6D75                   1191 :       HL=4  ADD HL,SP  (&[^SP])=HL
6D75                   1192 :       POP  HL
6D75                   1193 :       RET
6D75 18 2B             1194         JR &[GETREG_RET]
6D77                   1195 :
6D77                   1196 :       void  GETREG()
6D77                   1197 :
6D77                   1198 &[GETREG]
6D77 E5                1199         PUSH HL
6D78 FD 22 6E 6D       1200         (&[^IY])=IY
6D7C DD 22 6A 6D       1201         (&[^IX])=IX
6D80 22 66 6D          1202         (&[^HL])=HL
6D83 ED 53 63 6D       1203         (&[^DE])=DE
6D87 ED 43 60 6D       1204         (&[^BC])=BC
6D8B                   1205 :
6D8B F5 E1 22 7A 6E    1206         PUSH AF  POP HL  (&[^AF])=HL
6D90                   1207 :
6D90 21 00 00 30 01 23 1208         HL=0 IF C THEN INC HL
6D96 22 7F 6E          1209         (&[^CY])=HL
6D99                   1210 :
6D99 21 00 00 20 01 23 1211         HL=0 IF Z THEN INC HL
6D9F 22 81 6E          1212         (&[^ZERO])=HL
6DA2                   1213 :
6DA2                   1214 &[GETREG_RET]
6DA2 21 04 00 39 22 7D 6E 1215      HL=4  ADD HL,SP  (&[^SP])=HL
6DA9                   1216 :
6DA9 E1                1217         POP HL
6DAA C9                1218         RET
6DAB                   1219 :
6DAB                   1220 :
6DAB 21 40 6E C3 47 F0 1221 &[SIN]  HL=&TEMP JP #SIN+&SRB
6DB1 21 40 6E C3 4A F0 1222 &[COS]  HL=&TEMP JP #COS+&SRB
6DB7 21 40 6E C3 4D F0 1223 &[TAN]  HL=&TEMP JP #TAN+&SRB
6DBD 21 40 6E C3 50 F0 1224 &[ATN]  HL=&TEMP JP #ATN+&SRB
6DC3 21 40 6E C3 53 F0 1225 &[EXP]  HL=&TEMP JP #EXP+&SRB
6DC9 21 40 6E C3 56 F0 1226 &[LOG]  HL=&TEMP JP #LOG+&SRB
6DCF 21 40 6E C3 44 F0 1227 &[SQR]  HL=&TEMP JP #SQR+&SRB
6DD5 21 40 6E C3 5F F0 1228 &[RAD]  HL=&TEMP JP #RAD+&SRB
6DDB 21 40 6E C3 38 F0 1229 &[FIX]  HL=&TEMP JP #FIX+&SRB
6DE1 21 40 6E C3 3B F0 1230 &[INT]  HL=&TEMP JP #INT+&SRB
6DE7 21 40 6E C3 41 F0 1231 &[CINT] HL=&TEMP JP #CINT+&SRB
6DED 21 40 6E C3 3E F0 1232 &[FRAC] HL=&TEMP JP #FRAC+&SRB
6DF3                   1233 :
6DF3 CD F9 6D C3 59 F0 1234 &[POW]  &[GETARG2]  JP #POW+&SRB
6DF9                   1235 :
6DF9                   1236 :
6DF9                   1237 &[GETARG2]
6DF9 11 40 6E 26 00 3A AE 1238      DE=&TEMP  H=0 L=(#精度) ADD HL,DE
6E00 69 6F 19
6E03 EB                1239         EX DE,HL
6E04 C9                1240         RET
6E05                   1241 :
6E05                   1242 :
6E05                   1243 &[PUSH]
6E05 DD E1             1244         POP IX
6E07 06 04 3A AE 69 FE 05 1245      B=4  IF (#精度)=5 THEN  DEC B
6E0E 20 01 05
6E11                   1246         DO B {
6E11 56 23             1247           D=(HL)  INC HL
6E13 5E 23             1248           E=(HL)  INC HL
6E15 D5                1249           PUSH DE
6E16 10 F9             1250         }
6E18 DD E9             1251         JP (IX)
6E1A                   1252 :
6E1A                   1253 :
6E1A                   1254 &[POP]
6E1A DD E1             1255         POP IX
6E1C 3A AE 69 5F 16 00 19 1256      A=(#精度) E=A D=0  ADD HL,DE  DEC HL
6E23 2B
6E24 06 04             1257         B=4
6E26 FE 05 20 05       1258         IF A=5 THEN
6E2A D1                1259           POP DE
6E2B 72 2B             1260           (HL)=D  DEC HL
6E2D 06 02             1261           B=2
6E2F                   1262         FI
6E2F                   1263         DO B {
6E2F D1                1264           POP DE
6E30 73 2B             1265           (HL)=E  DEC HL
6E32 72 2B             1266           (HL)=D  DEC HL
6E34 10 F9             1267         }
6E36 DD E9             1268         JP (IX)
6E38                   1269 :
6E38                   1270 &WORK1:
6E38                   1271 :
6E38 33 E9             1272 &旧乱数   DW $E933
6E3A 31 32 33 34 35 00 1273 &WORK10   DM "12345",0
6E40 00 00 00 00 00 00 00 1274 &TEMP  DS 8*3 ;REAL型の引数は3個まで
6E47 00 00 00 00 00 00 00
6E4E 00 00 00 00 00 00 00
6E55 00 00 00
6E58 00 00 00 00 00 00 00 1275 &WORKREAL DS 34
6E5F 00 00 00 00 00 00 00
6E66 00 00 00 00 00 00 00
6E6D 00 00 00 00 00 00 00
6E74 00 00 00 00 00 00
6E7A 00                1276 &[^AF]    DB 0
6E7B 00 00             1277 &[^A]     DW $0000
6E7D 00 00             1278 &[^SP]    DW $0000
6E7F 00 00             1279 &[^CY]    DW $0000
6E81 00 00             1280 &[^ZERO]  DW $0000
6E83                   1281 :
6E83                   1282 &WORK1END:
6E83                   1283 :
6E83                   1284 &RUNTIME_END
6E83                   1285 :
6E83                   1286 ;****************************************
6E83                   1287 :
6E83                   1288         END
```

▶パソコンフォーラムに行ってロゴバッジをもらったけど，こんなのどうしろっていうんだ？　X68000PROにくっつけてもかっこ悪いし。　新田　弘樹(16)埼玉県

## 全機種共通システムインデックス



＊以上のアプリケーションは，基本システムであるS-OS"MACE"またはS-OS"SWORD"がないと動作しませんのでご注意ください。

INDEX

▶5月からMIDI音源を手に入れるため料理屋のアルバイトを始めました。しかし，X68000を触る時間が少なくなるし，疲れもたまって大変です。　後藤　秀樹(16)福島県

# マシン語カクテル in Z80's Bar

## 第23回——アフターケアなの?——

シナリオ：金子俊一
特別監修：浦川博之

今回も光君ががんばる「マシン語カクテル」。この酒場も人手不足のようですね。お話は，デバッガ作成の後日談と絡んでのZ80よもやま話です。プログラムはないんですが，前回，前々回，そして前々々回がわりと大きかったので，ごかんべんのほどを。

♪カラン，コロ～ン
**源光**（以下光）：カランカラン，コロン。
**ようこ**（以下Yo）：オッパケ～の。
**マスター**（以下M）：こらこら，鬼太郎の店にする気ですか。
**長老**（以下老）：ふぉっふぉっふぉっ。
**光**：出たな，子泣き爺い。
**老**：だれが子泣き爺いじゃ。
**光**：妖魔退散！
**老**：ぐえぇぇ。
**M**：長老もノリますね。
**Yo**：いったいどうしたの？
**光**：いや，遊びに来ただけなんですよ。
**老**：だからといってワシで遊ぶこたあないじゃろうに。
**光**：まあ，そういわずに。妖魔退散！
**老**：ぐえぇぇ。
**Yo**：光君，老人をもてあそぶもんじゃありません。
**光**：だって暇なんだもん。
**M**：光君も遊んでないで，来たついでになにかしていけば。
**Yo**：そうよ。世のため，人のため，ひいてはZ80's BARのためになにかしなさい。
**光**：う～ん。そういわれてもなあ。
**Yo**：なによ。私のいうことが聞けないの。
**光**：わかりましたよ，強引だなあ。
**M**：この前までやっていた，デバッガの話でもしてくださいよ。
**光**：え～，またですか。
**光**：そうですね。それでは今夜はZ80の生い立ちの話なんかでもしましょうか。

### 8080コンパチ

**光**：えっと，その昔に8080っていうインテル社のCPUがあったのを知ってる？
**Yo**：知らないわ。
**光**：今夜は素直ですね。
**Yo**：私を怒らせたいの？
**光**：そんなことはないです。えっと，某国民機と呼ばれるコンピュータに載っているCPUは知ってる？
**Yo**：えっと，80286とか80386ね。
**老**：時代は変わったのう。
**M**：変わりましたねえ。
**光**：その80286とか80386の前に8086っていうCPUがあったんですよ。
**Yo**：へぇ～。V30なら聞いたことあるんだけどな。
**光**：V30はいいんです。その8086は16ビットのCPUだったんですがね，さらに前に遡ると，8080っていう8ビットCPUがあったんですよ。
**Yo**：それがどうかしたの？
**光**：とっても重要なんですよ。その8080はZ80のモトなんですから。
**Yo**：Z80のモト？
**光**：Z80を作っているザイログ社というのは，もともとインテルで8080を作った開発メンバーが中心になって興した会社なんですよ。
**老**：そうそう，そうじゃったのう。
**光**：そこでZ80は全盛だった8080の上位コンパチCPUとして売り出された。
**Yo**：上位コンパチ？

表1　Z80になって追加された命令

| 8080の空きコード | 命令 |
|---|---|
| $08 | EX AF,AF' |
| $10 | DJNZ |
| $18 | JR e |
| $20 | JR NZ,e |
| $28 | JR Z,e |
| $30 | JR NC,e |
| $38 | JR C,e |
| $CB | ビット操作命令 |
| $D9 | EXX |
| $DD | IX系命令 |
| $ED | その他命令 |
| $FD | IY系命令 |

表2　Z80の命令形式

(1)
- OPコード

(2)
- OPコード
- オペランド

(3)
- OPコード
- オペランド（下位バイト）
- オペランド（上位バイト）

(4)
- 第1OPコード
- 第2OPコード

(5)
- 第1OPコード
- 第2OPコード
- ディスプレースメント

(6)
- 第1OPコード
- 第2OPコード
- オペランド（下位バイト）
- オペランド（上位バイト）

(7)
- 第1OPコード
- 第2OPコード
- ディスプレースメント
- オペランド

M：そういえば最近あんまり聞かなくなりましたね。
老：そういえば，そうじゃのう。
光：さっきからそればっかしですね。
Yo：そういえば，そうじゃのう。
老：こら，老人をからかうもんじゃない。
Yo：それで上位コンパチっていうのは？
光：コンパチっていうのは互換性があるっていうことですよ。
Yo：ふうむ。
光：そこで上位コンパチっていうのは，X1に対するX1turboみたいなもんで，X1turboではX1用のソフトは動くけど，X1ではX1turbo用のソフトは動かないっていうようなことだよ。
Yo：X68000みたいにどれも同じっていう場合はどうなるの？
光：それは完全コンパチってやつだね。
Yo：ふうん。
光：そこで話を戻すと，Z80は8080の上位コンパチだから，8080用のプログラムはちゃんと動くんだ。
Yo：じゃあ，Z80は8080になにかしらのものをつけたってわけね。
光：そういうこと。便利な命令がいっぱい増えているんだ。もともと8080では244種類しか命令がなかったので，1バイト命令しかなかったんだ。
Yo：それじゃ，先月話してた$CBで始まるビット操作命令もZ80のオリジナルなのね。
老：それだけではないぞ。$DDで始まるIX系の命令，$FDで始まるIY系の命令，みんなそうじゃ。
光：Z80になると命令数はなんと696種類にもなるんだ。
Yo：いっきに3倍近くも増えたのね。
老：実行可能な未定義命令を含めるともっと増えそうじゃのう。
Yo：実行可能な未定義命令？
老：そうじゃ。ザイログが認定してないけども，実際は動いてしまうマシンコードがあるんじゃ。
Yo：へえ～。
光：まっ，その話はまたいずれということにして，696種類もあると当然1バイトでは足りなくなるでしょ。
Yo：そうねえ。
光：そこで，ザイログは8080の空きコードを利用して，2バイト命令や新たな1バイト命令を作ったってわけさ。表1にまとめておくね。
Yo：あったまいいのねえ，感心しちゃう。
光：それが第1OPコードや第2OPコードの始まりってことだよ。

## 命令形式はちょっと複雑

Yo：ねえ光君，デバッガを作るうえで苦労したことってある？
光：そうだな，命令によってバイト数が違ったのはめんどくさかったな。
Yo：なにか規則性のようなものはないの。
光：そうだね，あればもっと楽だったんだけどな。
Yo：やっぱりZ80と8080では違うのよね。
光：うん，そうだね。表2を見て。これは命令形式の表なんだけど，8080なら(1)～(3)までですむんだ。
Yo：これってひとつの枠が1バイトってことね。
光：そう。8080では最長で3バイト命令ってことだよ。
老：MC68000では10バイト命令というのもあるそうじゃからな。かわいいもんじゃろ。
Yo：きれいに1,2,3バイトで並んでいる。
光：Z80ではさらに(4)～(7)までが加わるのでけっこう複雑になるでしょ。
Yo：4バイト命令までできるのね。
光：そうだね。

## 省メモプログラム

Yo：そういえばあのプログラムはちょうど4Kバイトに収まってたわよね。
光：そうだね。あれも苦労したな。4Kをずいぶんとオーバーしてたもんな。最初に作っていたアルゴリズムでは4Kをはるかにオーバーしちゃうからって，ほとんど書き換えたこともあったし。
Yo：なにかテクニックがあるんでしょ。
光：いろいろあるけど，かなりえげつないよ。
Yo：う～ん。可能なかぎりJR命令を使うなんてどう？
光：そうだね。JPに比べて1バイトは縮まるよね。当たり前のことだけど。
Yo：そのわりにはJP命令が多いように思えるんだけど。
光：それはね，こういうことですよ。

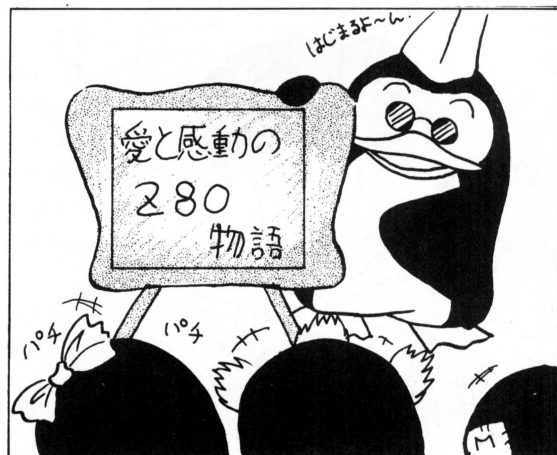
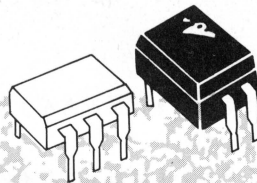
```
CALL ????
RET
```

このパターンを

```
JP   ????
```

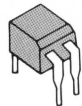
に書き換えたんですよ。
Yo：ちゃんと動作するの？
光：CALL先には必ずRETがあって，それで帰ってきますよね。
Yo：そうね。
光：そこでJPにしちゃえばスタックを使わないからスピードアップにもなるし，メモリも1バイト節約になるでしょ。
Yo：なるほどね。ほかにはどんなのがあるの。
光：いままでのは比較的せこかったでしょ。苦労して1バイトしか縮まらない。いちばん有効なのは同じようなことをしているルーチンを見つけてサブルーチン化してしまうことだろうね。これはいっきに数10バイトも縮まる可能性があるし，プログラムもまだまともに見える。
Yo：それらが積み重なるとスパゲッティになっていくのね。
光：そうだね，特に逆アセンブルのところは汚いね。その前はなんにも考えないでプログラム作ってたから。あとで後悔したよ，もっとタイトルを短くしておけばよかったとかね。
老：後悔先に立たずじゃ。
光：そうですね。1つひとつの命令のバイト数を確かめて，もっと短くならないかって必死でしたよ。
光：せっかく遊びに来たのになんだかZ80物語でも話に来たみたいだなあ。
Yo：まあいいじゃないの。それより，今度私をどこか連れていってね。
光：なんかどっかで聞いたことあるセリフだな。
Yo：わたしがいったのは初めてよ。さては，メアリーにでもいわれたんでしょ。
老：メアリーちゃんも見かけんのう。
光：純ちゃんや善ちゃんはどこいっちゃったのかなあ。
Yo：ごまかしてるわね。
M：山田君はツケが残っているのにな。

―つづく―

ハードウェア工作入門《13》

# メカトロニクス制御(その3)

Misawa Kazuhiko
三沢 和彦

**今月は，ステッピングモーターを使った秒カウント指針式時計プログラムを組みます。今回のリストはメカトロニクス制御の基本プログラム例。来月号になってしまった応用編までにちょっとした準備体操のつもりで読み進めていくといいでしょう。**

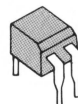
前回までのステッピングモーターの工作は完成しているでしょうか。トランジスタやダイオードなどの個別部品を取り付けるときに足を間違えると動作しないので，気をつけてください。今回は応用プログラミング編ですが，まずは動作チェックをかねてサンプルプログラムから試していきましょう。なお，あらかじめおわびしておきますが，今月の締め切りが筆者の海外出張と重なってしまったために，当初予定していたメカトロニクス制御までは間に合わせることができませんでした。今月はステッピングモーター単体だけで動かせる簡単なプログラムを載せて，応用は次回に回すことにします。

## マウスで動かす

さっそくリスト1から打ち込んでみてください。このプログラムでは，マウスの右クリックでモーターの右回転，左クリックで左回転を操作しています。打ち込み終わったら，ジョイスティックポートにケーブルを接続し，モーターの電力用電源を入れてからRUNさせます。マウスの左右クリックでモーターが回転するのが確認できたでしょうか。

では，プログラムの中身を順番に解説していきましょう。まず変数を定義したあとにタイマ用変数のnをセットします。インタプリタ上で動作させるときにはこのタイマルーチンtimer( )は必要ないのですが，コンパイルすると動作が速くなるので，この値で速度を調節します。とはいえ，コンパイルしたプログラムでタイマルーチンがなくてもモーターの応答が追いつかないということはないようです。

ところで，正確にモーターの回転速度を計ったことがないので，毎分何回転というように速度のかたちで設定することはしていません。もしも正確に速度を決めたいときはX68000内部のタイマで割り込みをかけてやらなければなりませんが，これは次回以降の課題とします。

そのあといくつかの初期設定をしてから，ループに入ります。ループの中ではmsstat関数でマウスの状態を読み取り，右ボタンが押されていたら，右回転のためのrightルーチンに，左ボタンが押されていたら，左回転のためのleftルーチンに飛びます。回転位置は絶対位置を示す変数psと，4相を順次ONしていくために相対位置を示す変数vとで管理しています。電磁石コイルは4相しかないので，vは0から3までの値を順次とります。そして，実際の回転には，outval関数でvの値をポートに出力することで行います。このoutval関数は最初の基本I/O回路のコントロールで10進数を出力するときに使った関数と基本的には同じです。前回は3ビット整数だったので，v0，v1，v2の3桁を使っていましたが，今回は0から3の2ビットなので，v0とv1のビットのみ使用し，v2ビットは先月述べたようにICのイネーブルに使っています。このプログラムでは常時イネーブルにしているので，v2は常に1にセットしています。

以上，リスト1を最初からていねいに追っていきましたが，実際にステッピングモーターを回転させるためにはoutval関数で0～3を順番に出力していくだけでできてしまうのです。

## 秒カウント指針式時計

理論編で，ステッピングモーターの身近な例のひとつとして指針式のデジタル時計を挙げましたが，ここでその実例を示してみます。リスト2を見てください。このプログラムはストップウォッチと同じように，スタートボタンを押すと計時を始め，もう一度押すと一時停止します。さらにもう一度押すと計時を再開し，また，停止しているときにリセットボタンを押すと針が元に戻るしくみになっています。実際には，スタート・ストップがマウスの左クリック，リセットが右クリックになっています。

このプログラムについても順番に説明していきましょう。まず，変数宣言のあと，毎秒ごとに進むステップ数をnに設定します。これは，1ステップが1.8°なので，毎秒1.8×n°進むことになります。さらに初期化後，スタートの左クリックを待つループに入ります。左クリックして離した瞬間に，“start”を表示して計時を開始します。メインループでは秒カウントしながら，左クリックを監視します。秒カウントのcountルーチン内では，左クリックを検出すると停止のためのフラグを立て，それによって，カウントループから抜けます。ループを回り続けるかぎり，stepルーチンでステッピングモーターを回し，また総経過秒数をsec変数に累計していきます。一時停止状態で右クリックを検出するとすべてのループから抜け，centerルーチンで指針を元に戻してからプログラムを終了します。このプログラムの基本はonestepルーチンで，リスト1とまったく同様に変数vを0から3まで順次進めながらoutvalで出力していくだけです。

今回使用したステッピングモーターは，1回転200ステップなので60では割り切れないため，残念ながら1周60秒のストップウォッチにはなりませんでした。毎秒2ス

テップと設定すると,100秒計ということになります。ここで，１秒で動かすステップ数をあまり欲張ると１秒以内で回り切らずに誤動作してしまいます。インタプリタ上では毎秒50ステップ程度が限度のようです。

しかし，毎秒50ステップとなると１周４秒計となり，あまり意味がなくなってしまいます。ここは，毎秒２ステップの100秒計か毎秒４ステップの50秒計がよいでしょう。

＊　　＊　　＊

いかがでしたか。以上，ステッピングモーターを動かす基本例をプログラムしてみました。来月は，模型にモーターを組み込んで，ロボットもどきの実験に挑戦してみるつもりですので，お楽しみに。

**リスト1**

```
 10 /* save "d:¥basic¥motor.bas
 20 /* save@"d:¥basic¥motor.doc
 30 /*
 40 /*ステッピングモータープログラム
 50 /*
 60 /*     1991.5.3  K. Misawa
 70 /*
 80 int v,n,x,y,bl,br,ps
 90 /*
100 /*タイマー用変数
110 n=1
120 /*
130 /*初期化
140 outval(4)
150 mouse(4)
160 /*
170 while 1
180   msstat(x,y,bl,br)
190   if bl=-1 then { v=left(v) } else {
200     if br=-1 then { v=right(v) }}
210    timer(n)
220    print ps
230 endwhile
240 end
250 /*
260 /*右回転
270 /*(引数)ONになっている相番号
280 /*(戻り値)1ステップ進んだ相番号
290 /*
300 func right(v;int)
310   v=v+1 : ps=ps+1
320   if v=4 then v=0
330   outval(v)
340   return(v)
350 endfunc
360 /*
370 /*左回転
380 /*(引数)ONになっている相番号
390 /*(戻り値)1ステップ進んだ相番号
400 /*
410 func left(v;int)
420   v=v-1 : ps=ps-1
430   if v=-1 then v=3
440   outval(v)
450   return(v)
460 endfunc
470 /*
480 /*データ出力
490 /*(引数)整数値
500 /*(戻り値)なし
510 /*
520 func outval(d0;int)
530   int v,v0,v1,v2
540   v0=1-(d0 and 1)
550   v1=1-(d0 and &B10)/&B10
560   v2=1
570   v=&B10000000*v1+&B1000000*v0+&B10000*v2
580   ioout(v)
590 endfunc
600 /*
610 /*回転速度制御タイマー
620 /*(引数)カウント数
630 /*(戻り値)なし
640 /*
650 func timer(n)
660   for iii=1 to n
670   next
680 endfunc
```

**リスト2**

```
 10 /* save "d:¥basic¥watch.bas
 20 /* save@"d:¥basic¥watch.doc
 30 /*
 40 /*ステッピングモーター応用
 50 /*  指針式時計プログラム
 60 /*
 70 /*     1991.5.4  K. Misawa
 80 /*
 90 int v,n,x,y,bl,bl0,br,sec,ps
100 str sm0
110 /*
120 /*毎秒ごとに進むステップ数
130 n=2
140 /*
150 /*初期化
160 outval(v)
170 mouse(4)
180 /*
190 /*スタート待ち
200 while (bl=0 and bl0=-1)=0
210   bl0=bl
220   msstat(x,y,bl,br)
230 endwhile
240 /*
250 sm0=time$
260 while time$=sm0
270 endwhile
280 cls : print"start"
290 /*
300 /*メインルーチンループ
310 while br=0
320 /*
330 /*秒カウント
340   bl=0 : bl0=0 : flg=0
350   while flg=0
360     print count(time$)
370   endwhile
380 /*
390 /*一時停止
400 /*  右クリックで終了、定位置へ
410   bl=0 : bl0=0
420   while (bl=0 and bl0=-1)=0
430     bl0=bl
440     msstat(x,y,bl,br)
450     if br=-1 then break
460   endwhile
470 endwhile
480 /*
490 center(ps)
500 end
510 /*
520 /*秒カウントルーチン
530 /*(引数)スタート時刻
540 /*(戻り値)経過時間(秒)
550 /*
560 func int count(tm0;str)
570   while time$=tm0
580     bl0=bl
590     msstat(x,y,bl,br)
600     if (bl=0 and bl0=-1)=-1 then flg=-1
610 /*   print time$
620   endwhile
630   sec=sec+1
640   step(n)
650   return(sec)
660 endfunc
670 /*
680 /*データ出力
690 /*(引数)整数値
700 /*(戻り値)なし
710 /*
720 func outval(d0;int)
730   int v,v0,v1,v2
740   v0=1-(d0 and 1)
750   v1=1-(d0 and &B10)/&B10
760   v2=1
770   v=&B10000000*v1+&B1000000*v0+&B10000*v2
780   ioout(v)
790 endfunc
800 /*
810 /*毎秒のステップ
820 /*(引数)毎秒進むステップ数
830 /*
840 func step(n;int)
850   for iii=1 to n
860     onestep()
870   next
880 endfunc
890 /*
900 /*1ステップ
910 /*
920 func onestep()
930   v=v+1 : ps=ps+1
940   if v=4 then v=0
950   outval(v)
960 endfunc
970 /*
980 /*定位置リターン
990 /*(引数)現在位置
1000 /*
1010 func center(ps;int)
1020   int rot,rest
1030   rot=ps¥200+1
1040   rest=200*rot-ps
1050   step(rest)
1060 endfunc
```

★(で)のショートプロぱーてぃ その22

# おお，グラフぃっく

Komura Satoshi 古村　聡

作者の努力により，今月からぱーてぃハンズの第3部を始めることができました。ネタはパズルゲーム。完成する日をお楽しみに。名前は「選択二十五」です。あっ，仮称と変わってないじゃないか……。

illustration : T. Takahashi

しっかし，なんですね。X68000ってのはにぎやかなマシンですね。私なんかは味もそっけもなく何の音も出ない，至極普通な立ち上がりのシステムにしてるんですが，人のマシンを触るとまあ，みんなにぎやか（やかましい？）なマシンに仕立て上げているのでびっくりしてしまいます。

たとえば，私の友人某の場合。某“管理人さん”が好きで，某“ユリ”が好きというこの人は，とあるPCMファイルを……，そう，もうわかるでしょ。

電源を入れると“がんばってくださいね”，コマンドまたはファイル名がみつからないと“残念でしたっ！”，ワープロを立ち上げると“ちょっと待ってくださいね”。……もうしゃべる，しゃべる。私が一度編集室で，何も知らずに彼のディスクを借りて立ち上げたら，マシン室中にフルボリュームで響子さんの“がんばってくださいね”の声が……。あのときはちょーーっとばっかしはずかしかったな，うん。じゃ，ブロンウィンやタムリンだったらいいのかって？　そ，それはいわない約束でしょ……。

はたまた，別の某氏の場合。この人はイースを見ると血がたぎり，リリアを見れば肉燃えるというお方。ワープロで原稿を書いている間中，ずっとイースの草原の音楽が流れています。仕事でもこの音楽がかかっているとディスプレイに向かう気が起こるんだとか。なるほど，さぞかし経験値も溜まることでありましょう。

ところが，この彼にもひとつ悩みがあるんだそうなのですね。

「どしたの？」

「うん，いっつも同じ音楽でしょ。せっかくOPMファイルもいっぱいあるんだから，日によって違う音楽になるようにできないかなあ。今日は草原，明日は洞窟，その次は塔の中みたいに……」

できますぜ，だんな。そんな貴方のための今月のショートプロ，なのであります。

## 日替わりぺけろく定食

というわけで，X68000用のちょっと使えるプログラムです。

ohayo.x for X68000

（要as.x,lk.xなど）

岩手県　大久保　明弘

Human68kにはバッチファイルというのがあるのはご存じですよね。そうそう，やりたいことをずらずらずらーっと書いておいて，それを一気に実行させるあれのことです。立ち上げ用のディスクなんかにもAUTOEXEC.BATなんてのが入ってて，電源を入れたり，リセットを押したときにこれに書いてあることが実行されるのも知ってますよね。

このバッチファイルを実行するときに，今日の日にちが知りたかったり，今月が何月か知りたいときってありませんか？　実はこのプログラムはそのためのプログラムなのです。

このプログラムの使い方なんですが，バッチファイル中で，

　ohayo　－m　（あるいは－d,－w）

とすると，－mで今月の月が，－d,－wで今日の日付け，曜日（1が日曜，2が月曜，……）が返ってきます。

たとえば，バッチファイルに，

```
ohayo －w
if exitcode 2 echo 今日は月曜だよ
```

とすれば，月曜日だったら「今日は月曜だよ」と画面に表示され，また，

```
ohayo －d
if　exitcode 18 copy　X68000.opm
opm
```

というふうに書いて，同じディレクトリに「X68000のテーマ」のOPMファイル（X68000.opm）を置いておくと，バッチが実行されたときにOh!Xの発売日であれば，「X68000のテーマ」が流れるわけです。

このプログラムリストはアセンブラのソースリストという形になっていますので，アセンブルしないとこのプログラムは実行できません。そこで一応，ざっとアセンブルの方法を説明しておきます。

このプログラムを使うには，Human68kで使えるエディタ，アセンブラ，リンカ，それにアセンブラ用のマクロであるiocscall.macとdoscall.mac（C compiler PRO-68KやOh!X 1月号の付録ディスクなどに入っています）が必要になります。ここではエディタにed.xとアセンブラ，リンカにas.x，lk.xを使う場合について説明しておきます。

まず，必要なアセンブラやマクロファイルなどをいま自分のいるディレクトリにコピーしてきてください。

まず，リストを打ち込みましょう。

　A>ed ohayo.s

と打ち込んでください。これでエディタが立ち上がりますのでリストを打ち込んでください（行番号は入れないように）。

打ち終わりましたか？　それじゃ，打ち込んだリストをセーブして，エディタを終わります。

　[esc] e

(ESCを押してからEを押す)してください。

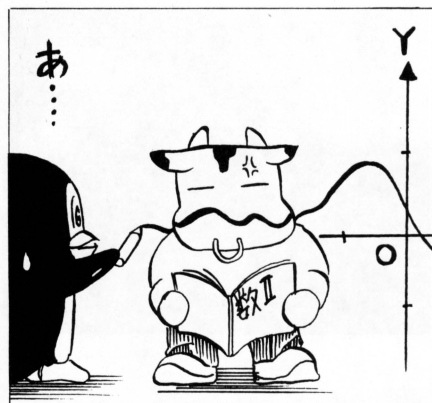
次は，打ち込んだリストをコマンドとして使えるようにする，アセンブルという作業です。

　A>as ohayo

と打って，それから

　A>lk ohayo

と打ってみてください。何もエラーは出ていませんね？　出ていたら，打ち間違いがありますからエディタで間違っているところを直してから，もう一度アセンブルしてください。

さて，これでohayo.xができました（dirでohayo.xがあることを確認してね）。サンプルを参考にバッチファイルをゴリゴリし

てください。さあ，これでX68000も日替わりメニューなのだ！

## あかでみっくX1

では続きましては今月の２本目。なんとまあ，式を入れただけで自動的にグラフを書いてくれるプログラムなのであります。

**ぐらふくん for X1turbo**

(turboBASIC)

**東京都　木村　哲也**

まず，プログラムを入れます。間違えないようにね。RUNするとメニューが表示されます。

1)　ごく普通のy＝f(x)の形の式ですね。たとえばy＝2x＋3だったら，2＊x＋3と入力します。

2)　tを媒介変数としたX,Yに関する式ですね。たとえばx＝t,y＝2tのときは，順にt,2tと入力してください(もっともこれはy＝2xでもいいわけですけど)。

3)　x＝f(y)の形になっているものですね。たとえば$x=y^2$(入れるときはx＝y^2)とか。

4)　1)と同じ形の式ですが，赤で１回微分した式のグラフも一緒に書いていきます。

5)　r＝f(θ)の極座標表現式で，原点からrの距離のところに点をプロットしていきます。たとえば１を入れれば，原点(0,0)から半径１の円を，sinθであれば(0,0.5)から半径0.5の円を書きます。キーボードで式を入れるときにはθは使えないので，代わりにXを使ってください。たとえばsinθであれば，sin(x)と入れます。

6)　グラフをプリンタでハードコピー。

こ，こいつはすっごいですね。２次元だったらほとんどどんなグラフも描いてしまうんですねえ。sin,cos,log xどころかアークサイン，アークコサイン，ハイパボリックの三角関数まである……。

式としては，

r＝5cos(4x)

を標準画面モードでなんて，おすすめですよん。

私なんか，私なんか，高校でも数学なんか，微分積分なんか赤点取りまくって，それにもかかわらず大学は数学が専攻。sin,cos,tanには個人的に**ものすごーく**恨みがあるのに……。なんでこんなに**簡単に**グラフが描けちまうんだよう！　うるうるうるうるっ(興奮のあまり文章が支離滅裂)。

にしても99行でこんなプログラムが書けてしまうっていうのはすごいですね。投稿

### リスト1　ohayo.s

```
 1: *
 2: * おはようプログラム
 3: *          Written by Azuron 1991 3.22(fri)
 4: *
 5:            .include        iocscall.mac
 6:            .include        doscall.mac
 7:
 8: SPACE      equu     $20
 9: TAB        equ      $09
10:
11:            .text
12:            .even
13:
14:            tst.b    (a2)+                       *コマンドラインに何も入力してない
15:            beq      usage                       *  場合は使用法を表示して終わり
16:
17:            bsr      skipsp
18:            cmpi.b   #'/',(a2)                   *コマンドラインの先頭が'/'か'-'か調べる
19:            beq      swok
20:            cmpi.b   #'-',(a2)
21:            beq      swok
22:
23:            beq      usage                       *'/'か'-'でない場合は使用法を表示して終わり
24: swok:
25:            IOCS     _DATEGET                    *d0.l='0WYYMMDD'
26:            move.l   d0,d1
27:            IOCS     _DATEBIN                    *d0.l='WYYYMMDD'
28: swchk:
29:            addq.l   #1,a2                       *スイッチを調べる
30:            cmpi.b   #'M',(a2)
31:            beq      month
32:            cmpi.b   #'m',(a2)
33:            beq      month
34:            cmpi.b   #'W',(a2)
35:            beq      week
36:            cmpi.b   #'w',(a2)
37:            beq      week
38:            cmpi.b   #'D',(a2)
39:            beq      day
40:            cmpi.b   #'d',(a2)
41:            beq      day
42:
43:            bra      usage                       *'m','w','d'以外なら使用法を表示して終わり
44:
45: month:                                          *月の場合
46:            ror.l    #8,d0                       *WYYYMMDDをDDWYYYMMにする
47:            andi.l   #$0000_00ff,d0              *月だけを残してマスク
48:            bra      final
49:
50: week:
51:            rol.l    #4,d0                       *WYYYMMDDをYYYMMDDWにする
52:            andi.l   #$0000_000f,d0              *曜日だけを残してマスク
53:            addq.l   #1,d0                       *日曜日の場合終了コードを1にするため
54:            bra      final
55:
56: day:
57:            andi.l   #$000000ff,d0               *日だけを残してマスク
58:
59: final:
60:            move.w   d0,-(sp)                    *終了コードを指定して終わり
61:            DOS      _EXIT2
62:
63: *
64: skpsp0:                                         *スペース&タブをスキップ
65:            addq.l   #1,a2
66: skipsp:
67:            cmpi.b   #SPACE,(a2)
68:            beq      skpsp0
69:            cmpi.b   #TAB,(a2)
70:            beq      skpsp0
71:            rts
72:
73: usage:
74:            pea.l    message(pc)                 *使用法を表示して終了する
75:            DOS      _PRINT
76:            addq.l   #4,sp
77:            moveq.l  #0,d0
78:            bra      final
79:
80:            .data
81:            .even
82:
83: message:
84:            dc.b     '機能:月、日または曜日を終了コードに持ってプログラムを終了します',13,10
85:            dc.b     '使用法:ohayo [スイッチ]',13,10
86:            dc.b     '                       /m      月',13,10
87:            dc.b     '                       /w    曜日',13,10
88:            dc.b     '                       /d      日',13,10,0
89:            .even
90:
91:            .end
92:
```

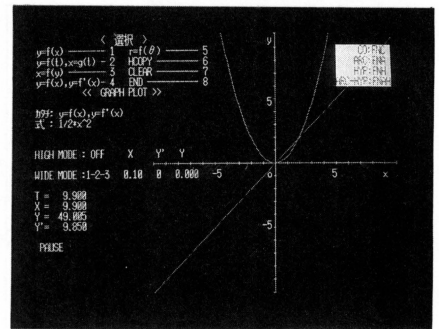
### リスト2　ohayo.xのサンプル

```
echo off
ohayo -d
if exitcode 18 echo  今日はOh！Xの発売日です。

ohayo -w
if exitcode 1    echo 今日はスピリッツとジャンプの発売日です。
if exitcode 2    echo 今日は火曜日です。
if exitcode 3    echo 今日はサンデーの発売日です。
if exitcode 4    echo 今日はヤンジャンとモーニングの発売日です。
if exitcode 5    echo 今日は金曜日です。
if exitcode 6    echo 今日は日本昔話がTVであります。
if exitcode 7    echo 今日は休息日です。のんびりいきましょ！

echo on
```

原稿によれば，このプログラムのミソはCALC命令なんだそうですが，さすがturboBASICです。奥の深さでは32ビットマシンにもひけをとらない。

とにかくX1turboを持っている人は何も考えずに打ち込んで，思いつくままにグラフを描いてみましょう。数学に苦しんでいる人は「おお，この式はこんなグラフになるのか」とか，むかし苦しんだ人は「ああ，このグラフには苦しめられたなぁ，なつかしいなあ」とか，思うこと請け合いです。

ただし，くれぐれも数学の宿題をこれですましてしまったり，あまつさえX1turboを学校に持ち込んでカンニングなんて暴挙に出ないように（でないって，そんなもん）。そんなことしてると，こういう大人になっちゃうぞお。こわいでしょ。

と，みじめな気分になったところでまた来月。このOh!Xでお会いしましょう。じゃんじゃん。

**リスト3　ぐらふくん**

```
100 '------ぐらふくん------------------------ BY 木村  哲也 --
110 '-----------------------------------------------------------
120 CLEAR:CLICK OFF:KEYLIST0:GOSUB 890
130 GOSUB 690:GOSUB 820
140 CONSOLE:LOCATE20,4:A$=INKEY$(1):A1=ASC(A$)-48:IF (A1>0)*(A1<
9)<>1 THEN140
150 PRINT:PRINT"                << GRAPH PLOT >>           "
160 LOCATE2,7:ON A1 GOTO 180,190,200,210,220,230,130,170
170  CONSOLE:CLS4 :PRINT"THANK YOU!":KEYLIST1:END
180  PRINT"ｶﾀﾁ: y=f(x)          ":INPUT" f(x)";FF$:GOTO250
190  PRINT"ｶﾀﾁ: y=f(t),x=g(t) ":INPUT" f(t)";FF$:INPUT" g(t)";GG
$:GOTO250
200  PRINT"ｶﾀﾁ: x=f(y)          ":INPUT" f(y)";FF$:GOTO250
210  PRINT"ｶﾀﾁ: y=f(x),y=f'(x)":INPUT" f(x)";FF$:GOTO250
220  PRINT"ｶﾀﾁ: r=f( θ ) ( θ =x)":INPUT" f(x)";FF$:GOTO250
230  HCOPY 0:GOTO 140
240 '------------  GRAPH             ------------------------
250 LOCATE 1,8 :PRINT" 式 :";:LOCATE1,9:PRINT"      "
260 A=.1 :B= 10 :M=13:LOCATE 2,11:PRINT"HIGH MODE : ON/OFF"; :A$
=INKEY$(1)
270 LOCATE14,11:IF A$="Y" THEN PRINT"ON     ":A=.01 ELSE IF A$<>"
" THEN PRINT"OFF    "
280 LOCATE 2,13 :PRINT"WIDE MODE :1-2-3 "; :A$=INKEY$(1)
290 IF A$="3" THEN B=20:A=A*10
300 IF A$="1" THEN B=2 :A=A/25
310 BB=B/2:GOSUB780
320 CONSOLE8,15,1,37:DY=999:ON ERROR GOTO 590
330 DEFFNY(X) =CALC(FF$)
340 DEFFNXT(T)=CALC(FF$):DEFFNYT(T)=CALC(GG$)
350 DEFFNIN(X,Y)=CALC(FF$)
360  X0=-B:ON A1 GOSUB370,380,390,410,400:GOTO420
370     Y0=FNY(-B)              :RETURN
380     Y0=FNYT(-B):X0=FNXT(-B):RETURN
390     Y0=FNY(-B) :SWAP X0,Y0 :RETURN
400     X0=FNY(-B)*COS(-B):Y0=FNY(-B)*SIN(-B):RETURN
410     Y0=FNY(-B) :LOCATE18,11:PRINT"    X      Y'   Y     ":RETU
RN
420 FOR T=-B TO B STEP .1
430  IF INKEY$<>"" THEN GOSUB 600
440  ON A1 GOSUB 450,460,470,490,480:GOTO 500
450     Y=FNY(T) :X=T       :RETURN
460     Y=FNYT(T):X=FNXT(T):RETURN
470     X=FNY(T) :Y=T       :RETURN
480     X=FNY(T)*COS(T):Y=FNY(T)*SIN(T):RETURN
490     GOSUB 450:GOSUB620 :RETURN
500   IF ABS(Y0-Y)>B   THEN Y0=Y
510   IF ABS(X0-X)>B   THEN X0=X
520 IF ABS(Y)< 1000 LINE(X0,Y0)-(X,Y),PSET,4
530  X0=X:Y0=Y:DY1=DY
540    LOCATE 2,15:PRINTUSING"T = ###.###";T
550    LOCATE 2,16:PRINTUSING"X = ###.###";X
560  LOCATE2,17:IF ABS(Y)<1000  THEN PRINTUSING"Y =####.###";Y E
LSEPRINT"Y = Error !"
570  LOCATE2,18:IF ABS(DY)<1000 THENPRINTUSING"Y'=####.###";DY E
LSEPRINT"Y'= Error !"
580 NEXT :GOSUB 600 :RUN
590 Y=1001: RESUME NEXT
600 K$=INKEY$:IF K$="" THEN:CFLASH 1:LOCATE 3,20:PRINT"PAUSE";:G
OTO600 'PAUSE
610 CFLASH:LOCATE 3,20 :PRINT"      ";:IF K$="B"THENCLS:GOTO140 E
LSE RETURN
620 IF X0-X<>0 THENDY=(Y0-Y)/(X0-X):IFABS(DY)<1000 THENPSET(X,DY
,3) ELSE RETURN  '-----ZOUGEN
630 IF DY*DY1<0 AND ABS(DY-DY1)<10 THEN LOCATE18,M:PRINTUSING" #
##.##   0";X;:PRINTUSING"####.### ";Y0:M=M+1
640 IF (Y*Y0<0)*(ABS(Y-Y0)<10)=1 OR Y=0 THENLOCATE18,M:PRINTUSIN
G" ###.##     ";X;:PRINT"    0     ":M=M+1
650 IF M=21 THENM=13
660 'DYY=(DY1-DY)/(X0-X):IF ABS(DYY)<1000THENPSET(X,DYY,2) '  F'
'(x)
670 RETURN
680 '-----------  GAMEN   SUB        ------------------------
690 OPTIONSCREEN0:WIDTH80,25,1,2:INIT
700 WINDOW(219,0)-(639,399),(-10,10)-(10,-10)
710 LINE(10,0)-(-10,0),PSET,1:LINE(0,10)-(0,-10),PSET,1
720 FOR C=-10 TO 10:LINE(C,.1)-(C,-.1),PSET,1:NEXT
730 FOR C=-10 TO 10:LINE(.05,C)-(-.05,C),PSET,1:NEXT
740 CREV1
750 LOCATE67,1:PRINT"     CO:FNC ":LOCATE67,2:PRINT"    ARC:FNA
"
760 LOCATE67,3:PRINT"    HYP:FNH ":LOCATE67,4:PRINT"ARC-HYP:FNAH
"
770 CREV:RETURN
780 WINDOW(219,0)-(639,399),(-B,B)-(B,-B)
790 CONSOLE:LOCATE40,13:PRINT-BB;"          o              ";BB;"
    x";
800 LOCATE52, 0:PRINT"y";BB;"";-BB:RETURN
810 '----------- SELSECT GAMEN       -------------------------
820 SCREEN0,0,0:CONSOLE
830 LOCATE0,0 :PRINT"                  <  選択  >
840 PRINT"  y=f(x) -------- 1   r=f( θ ) ------- 5
850 PRINT"  y=f(t),x=g(t) - 2   HCOPY --------- 6
860 PRINT"  x=f(y) -------- 3   CLEAR --------- 7
870 PRINT"  y=f(x),y=f'(x)- 4   END ----------- 8
880 PRINT"                                           ";:RETURN
890 DEF FNASIN(X)=ATN(X/SQR(1-X^2)   :DEF FNACOS(X)=-ATN(X/SQR(1
-X^2))+π/2
900 DEF FNASEC(X)=ATN(SQR(X^2-1))+(SGN(X)-1)*π/2:DEF FNSEC(X)=1/
COS(X)
910 DEF FNACSEC(X)=ATN(1/SQR(X^2-1))+(SGN(X)-1)*π/2:DEF FNACOT(X
)=-ATN(X)+π/2
920 DEF FNHSIN(X)=(EXP(X)-EXP(-X))/2 :DEF FNHCOS(X)=(EXP(X)+EXP(
-X))/2
930 DEF FNHTAN(X)=-EXP(-X)/(EXP(X)+EXP(-X))*2+1:DEF FNCSEC(X)=1/
SIN(X)
940 DEF FNHSEC(X)=2/(EXP(X)+EXP(-X)) :DEF FNHCSEC(X)=2/(EXP(X)-E
XP(-X))
950 DEF FNHCOT(X)=EXP(-X)/EXP(X)-EXP(-X))*2+1:DEF FNCTAN(X)=1/TA
N(X)
960 DEF FNAHSIN(X)=LOG(X+SQR(X^2+1)) :DEF FNAHCOS(X)=LOG(X+SQR(X
^2-1))
970 DEF FNAHTAN(X)=LOG((1+X)/(1-X))/2:DEF FNAHSEC(X)=LOG((SQR(1-
X^2)+1)/X)
980 DEF FNAHCSEC(X)=LOG(SGN(X)*SQR(X^2)+1)+1)/X)
990 DEF FNAHCOT(X)=LOG((X+1)/(X-1))/2  :RETURN
```

# （で）のぱーてぃハンズ第3部——（その1）

はっはっはっ。やっと今月からぱーてぃハンズ再開でいっ，というわけであります。予定が未定にならないでよかったなあ。しみじみ。

さてさて，今月のプログラムは思考型対戦ゲームを作って，その相手をコンピュータにやらせるというものなのであります。

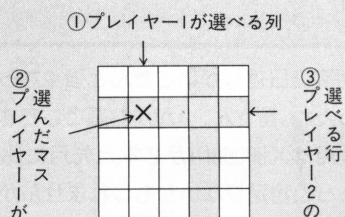
コンピュータになにがしかのことを考えさせるわけね。わーい，人工知能だ！　なんて大風呂敷を広げてしまう私なのです。

まあ，最近は人工知能というとナレッジエンジニアリングだ，ニューロだファジィだなんていうことになっているけど，コンピュータにゲームの手を考えさせる理論になるゲーム理論だってかつては人工知能の研究のひとつだったんですからね。だからゲームの思考ルーチンが人工知能入門の基礎だというのもあながちホラとはいえないのです。洗濯することがニューロだ，ファジィだ，バイオだっていうこのご時世。いいじゃないか，多少の大風呂敷は……。

能書きはさておき，パソコンの思考ルーチンを作りたい人にはいい題材になるんじゃないかと思います。

## 注意1秒，負け1回

話がそれたところで，このゲームについての解説です。実はこのゲーム，荻窪圭さんから「Macintoshにこういうソフトがあったよーん」というのを聞いて，よし作ってみようと作ってみたところ，何をどう聞き違えたのか全然別なものができてしまった，という実に私らしい偶然の産物オリジナルであったりするのですね。よって，きっと誰も見たことのないゲームなの

①プレイヤー1が選べる列
②プレイヤー1が選んだマス
③プレイヤー2の選べる行

で，心して聞くように。

まず材料は，

縦5×横5＝25個のマス
−9〜16までの25個の数字
プレイヤー2人

です。あ，あと，2人のスコアですね。最初は両方とも0点からスタートします。

縦5個，横5個，合計25個のマスがあります。これに−9から16までの数を乱数で入れていきます(画面写真を見てね)。で，プレイヤー1を先攻，2を後攻と決めて，最初の〝列〞を0〜4のうちから乱数で決めます。

ではまず，プレイヤー1のプレイです。先ほど，プレイヤー1は乱数で決めた〝列〞の5つの数字の中からひとつ数字を選びます。これをプレイヤー1のスコアに足して，選んだマスには×をつけます。

次はプレイヤー2の番です。プレイヤー2は先ほどプレイヤー1が選んだ数字のある“行”(わかると思うけどプレイヤー1が選べるのは縦1列の中から。プレイヤー2は横1行の中からになるわけね)の中からひとつ選びます。さっきプレイヤー1が選んだところはもう×がついていて選べないので，残り4つの中から選ぶことになりますね。選んだらその数字をスコアに足して，マスに×を描きます。そして，プレイヤー1がさっきプレイヤー2の選んだマスのある“列”で，×のついてないものを選び……，と繰り返していきます。プレイヤー1，2どちらかが取れるマスのなくなった時点でゲームは終わり。で，終わった時点で点数の多かったほうの勝ち，ということになるわけです。

文章で書くとなんか面倒臭くてわかりにくいけど，やってみると非常に単純なルールです。図を見て実際に自分でゲームをやってみてください(1人で2プレイっていうのもなんかむなしいけどね)。

## 小さな1歩は画面表示

さて，プログラムの説明に入ります。例によってハンズのプログラムは毎月，リストのうちいくつかの部分が掲載され，最終的にすべてのプログラムを組み合わせるとゲームが完成するという形を取っています。

それに加えて今回は少し変わって，その部分を打ち込むだけでその部分だけ実行して試すことができるようになっています。

そのことに伴って，前の号に出ていた行が次の号で少し変更になって出てくることもあります。ですから，何カ月かまとめて打ち込むときには掲載された順番どおり打ち込むように注意してください。

それと行番号についてなのですが，今回の第3部にかぎってgotoを使ってしまっています。その行が出てきた時点でまた注意しますが，renumコマンドには十分注意してください(全部完成した時点でちゃんと10行ごとになっているとは思うんだけどね)。

では，プログラムの解説に入りましょう。今月号掲載のプログラムは表示部分です。

今月のルーチンは，

| | |
|---|---|
| メイン： | initscrn( )の呼び出し |
| initscrn( ): | 画面を消す |
| | drawetc( )を呼ぶ |
| | drawbox( )を呼ぶ |
| | setnumbers( )を呼ぶ |
| | drawnumbers( )を呼ぶ |
| drawetc( ): | スコアetcを書く |
| drawbox( ): | マス目を描く |
| setnumbers( ): | マスに数字をランダムに置く |
| drawnumbers( ): | マスの数字を画面に書く |

という具合になっています。メインルーチンもその下のinitscrn( )もその下請けのルーチンに仕事を出しているだけなんですね。

私なんかがいえた義理ではありませんが，なるべく，ひとつの仕事はひとつの小さなルーチンに分けて別々に呼び出すようにしたほうがいいですよ。そうすれば1つひとつルーチンを呼ぶだけでプログラムのどの部分がまずいのか，だいたいの見当がつきますから。

大きいプログラムになればなるほど，プログラムのどの部分が間違えているかわかりにくくなりますからね。ほかのBASICしか知らない人や，初めてプログラムを組む人は使い方がまだわからない人もいるかもしれないですけど，マニュアルのfunc〜endfuncのところをよーく見て憶えておきましょうね。

プログラム自体は別にむずかしいところはないと思いますが，多少は解説を。

まず数字の入るマスなのですが，これにはint型の1次元配列をaryという名前で25個用意しています。これは1行目の左はしから配列aryの0番(ary(0))，1番，……，4番，次に2行目に行って5，6，……，という具合で最後がary(24)になるように並んでいます。

そして，setnumbers( )というルーチンでこの配列にランダムに数字を入れています。ここではtblという配列を用意して，マスに数字を入れるときにその数字がすでに使われていないかどうかチェックしています。

さて，新装開店したぱーてぃハンズ，いかがですか？　リスト中の注釈が少ないとか，もう少しリストの解説を増やして，などのハガキもあったようなので，少し感じを変えてみました。皆さんの感想を待っています。

では，また来月，このOh!Xで。よいしょっと！(と，気合いを入れて，去る)

★　選択二十五　★

リスト

```
1000 /*選択二十五                                    */
1010 /*                    by (で) 1991               */
1020 /*                    Special Thanks to荻窪sanなのだ*/
1030 /*                                              */
1040 /*    メインルーチン  */
1050    /*変数宣言*/
1060   dim int  ary(24)        /*枠の中身*/
1080   int      msgy=26        /*メッセージ出力y座標*/
1100   int      scr1=0,scr2=0  /*プレイヤー1，2の  スコア*/
1160   str      strbuf         /*文字列が一時入るバッファ*/
1180 /*                                  */
1190 /*------  ここからプログラム  ----------  */
1200 /*                                  */
1210       initscrn()
1520      end
1530 /* 画面の初期化サブルーチン*/
1540 func    initscrn()
1550   /*画面を消す*/
1560    width 96:screen 2,0,1,1
1570   /*その他の描画*/
1580   drawetc()
1590   /*マス目を描く*/
1600   drawbox()
1610   /*数を適当に配置*/
1620   setnumbers()
1630   drawnumbers()
1640 endfunc
1650 /*マスを描くルーチン*/
1660 func drawbox()
1670 int x,y
1680     for x=0 to 4
1690         for y=0 to 4
1700             box(x*64+96,y*64+64,x*64+96+63,y*64+64+63,15)
1710         next
1720     next
1730 endfunc
1740 /*乱数でマスを埋める*/
1750 func setnumbers()
1760 int i,j,tbl(24)
1770   locate 36,26:print"しばらくおまちください"
1780   /*乱数の使用表を作る*/
1790           for i=0 to 24:tbl(i)=1:next
1800   /*乱数を決める*/
1810           for i=0 to 24
1820                   repeat
1830                           ary(i)=int(rnd()*25)
1840                   until(tbl(ary(i))<>0)
1850                   tbl(ary(i))=0
1860           next
1870   endfunc
1880 /*ナンバーを画面に書く*/
1890 func drawnumbers()
1900 int x,y,i
1910   for x=0 to 4
1920           for y=0 to 4
1930                   locprn(x,y,15)
1940           next
1950 next
1960 endfunc
1970 /*画面のかざり*/
1980 func drawetc()
1990   symbol(148,0,"★    選択二十五    ★",2,2,2,15,0)
2000    drawscore()
2010 endfunc
2020 /*スコアを書く*/
2030 func drawscore()
2040    locate 64,9:print"PLAYER1"
2050    locate 80,9:print scr1
2060    locate 64,15:print"PLAYER2"
2070    locate 80,15:print scr2
2080 endfunc
2330   /*座標付きの数字プリント*/
2340 func locprn(locx,locy,col)
2350                   i=ary(locy*5+locx)-9:if i>=0 and i<=9 the
n{
2360                           symbol(locx*64+96+16,locy*64+64,
strs(i),2,2,2,col,0)
2370                   }else if i>-10000 then{
2380                           symbol(locx*64+96,locy*64+64,str
s(i),2,2,2,col,0)
2390                           }else symbol(locx*64+96,locy*64+64
,"×",3,3,2,5,0)
2400  endfunc
```

## X68000用(ZMUSIC.FNC要)
# 今すぐKISS ME

Sakai tohru
酒井 徹

## X68000用
# 歩いていこう

Ishigami Satoshi
石神 覚司

初夏の日差しがいちだんと強くなってきましたね。皆さん，いかがお過ごしですか？ 今月はX68000用が2本と先月に比べてちょっと物足りないかもしれませんが，その分元気な曲をお届けします。以前に予告していたゲームミュージック特集は来月号に決定しました。ゲームミュージックファンの人はもうしばらく我慢していてください。

### 世界でいちばん君が好き

まず，1曲目はX68000，OPMD&ZMUSIC.FNC用の「今すぐKISS ME」です。この曲は，「安定した視聴率をかせぐ」といわれる浅野温子を起用したフジテレビ系の人気ドラマ，「世界でいちばん君が好き」の主題歌にもなっていたので，わりと有名だと思います。歌っているのはLINDBERGです。LINDBERGは，このLIVE inにも3月号に続いて2度目の登場ですね。

余談ですが，この「今すぐKISS ME」は，アルバム「LINDBERGⅢ」の3曲目に収録されているのですが，以前掲載した「LITTLE WING」も，同アルバムの1曲目に収録されています。ちまたではすでに「LINDBERGⅣ」もリリースされていますが，有名曲2つを収録したアルバムということで，「〜Ⅲ」もまだまだ人気の高いアルバムです。

ところで，この曲はZMUSIC.FNCを使って作られていますので，ZMUSIC.FNCをお持ちでない方は演奏できません。ちょっと問題ありです。ご注意ください。

さて，この曲を演奏させるためには，まずバッファの確保が必要です。標準のシステムではOPMDRV側のバッファが64Kバイトしか確保されていないため，バッファが確保できないのです。CONFIG.SYSファイルをエディタで呼び出して，OPMDRVの行を以下のように書き換えましょう。

```
DEVICE = ¥SYS¥OPMDRV.X #256
```

これですべてのトラックをあわせて256Kバイトを確保しています。平均して1トラックあたり32Kバイトのバッファを取れるようになります。もちろん，“m-alloc”を個別で使えば，1トラックだけ200Kバイトを使うこともできますが……。このシステムではOPMDRVを使わなくても256Kバイトをメモリ上にとってしまうので，あまり効率がよいとはいえません。この際いい機会ですから，音楽専用のシステムを作ることをお勧めします。

LINDBERG

この作品では，ギターのディストーションがかかりすぎなのか，かなりひずんでますね。間奏の部分ではちょっとうるさいかな，と感じます。その反面，ヴォーカルはきれいになりすぎています。ヴォーカルの渡瀬マキさんは，無表情ながら腹式呼吸バリバリの比較的パワフルな歌い方をする人ですので，音のイメージが違うといえます。研究してみてください。それからヴォーカル全体にエコーが残っていて，カラオケを聴いているみたいです。切るところは切ってメリハリをつけましょう。

それから一度目の「ドキドキすること〜♪」のコーラスギター「すること〜♪」がハズレてるように聞こえませんか。もう少しスマートでもいいでしょう。

とはいえ，全体的に見るとOKな作品です。打ち込んでみましょう。

### J(S)W初登場!

もう1曲はJUN SKY WALKER (S) の「歩いていこう」をお届けしましょう。X68000，OPMA用です。バンドブームももう終わりといわれていますが，先頃発売されたアルバム「START」は，オリコン第1位という売れ行きぶりのJ (S) W。この曲は彼らの2ndアルバム「歩いていこう」の1曲目です。このアルバム自体はそれほどメジャーではないかもしれませんね。この曲だけはメジャーなんですけどね。

JUN SKY WALKER(S)

ところで，OPMDとOPMAはMIDIを除くとコンパチになっています。特にMIDIと書いてなければ，どちらでも演奏することは可能ですが，両者の大きな違いはサンプリングデータにあるのです。OPMAは電波新聞社から発売されているボスコニアンのサンプリングデータを使います。OPMDはOh!Xのオリジナル・サンプリングデータを使います。同じような音を選んではありますが，やはり違った音になってしまった部分もあり，投稿者がどちらの音を基準に作ったのかによって微妙に変わってくるのです。そこで，OPMA用やOPMD用というのは，どちらのサンプリングデータを使えばいいのかという目安にしてください。OPMDしか持っていないのでOPMA用は鳴らない，ということはありませんし，どっちもないという人でも演奏はできますので念のため。

さて，作者の石神君は，この曲をT-SQUAREの「TRUTH」といっしょに投稿

してくれました。どうやら「TRUTH」のほうがメインで，「歩いていこう」はオマケだったらしいのですが，なぜかこちらが採用されてしまいました。原曲のイメージよりも，ちょっとのんびりした感じがいいですね。さすがに初めて作ったMMLということで，仕上がり具合はイマイチですが，そのイマイチなところが高校の文化祭のようで，また捨てがたいものがあります。

石神君のこれからの課題としては，全体的に音の厚みが少ない点をもっと勉強してほしいですね。いろいろ試したりして研究してみてください。

バンドミュージックを送ってきてくれた2人は奇しくも同じく17歳。これからが楽しみですね。今年は高校3年生ということで，大変な時期を迎えていることでしょう。頑張ってくださいね。　(S.K.)

## リスト1　今すぐKISS ME



```
 10 /*******************************************************/
 20 /***                                                 ***/
 30 /***                                                 ***/
 40 /***    今すぐKissMe         LINDBERG              ***/
 50 /***                                                 ***/
 60 /***                                                 ***/
 70 /***        P r o g r a m e d   B y   酒井  徹      ***/
 80 /***                                                 ***/
 90 /***                                                 ***/
100 /*******************************************************/
110 m_init():for i=1 to 8:m_alloc(i,19000):m_assign(i,i):next
120 dim str p(99)[256]:char o(255)
130 /*******************************************************/
140 /***        音 色 設 定                              ***/
150 /*******************************************************/
160 dim char v(4,10)={
170 /*  AF  OM  WF  SY  SP PMD AMD PMS AMS PAN
180     59, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
190 /*  AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME  E.Bass
200     31, 12,  5,  6, 13, 35,  2,  5,  0,  0,  0,
210     31, 10,  5,  6, 13, 20,  0,  0,  0,  0,  0,
220     31, 10,  5,  6, 13, 40,  0,  0,  0,  0,  0,
230     31, 10,  5,  7, 13,  2,  0,  1,  0,  0,  0}
240 m_vset(70,v)
250 /*
260 v={
270 /*  AF  OM  WF  SY  SP PMD AMD PMS AMS PAN
280     41, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
290 /*  AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME  Hi-Hat Op
300     31, 31,  2, 15,  1,  7,  0, 11,  0,  3,  0,
310     31, 28,  0, 15,  4, 16,  2, 12,  0,  3,  0,
320     31, 22,  2, 15,  4, 10,  1,  1,  0,  1,  0,
330     31, 31,  8, 15,  0,  0,  2,  7,  0,  0,  0}
340 m_vset(71,v)
350 /*
360 v={
370 /*  AF  OM  WF  SY  SP PMD AMD PMS AMS PAN
380     60, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
390 /*  AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME  Bass Drum
400     31,  0,  0,  8,  0,  0,  0, 15,  0,  3,  0,
410     31, 19, 12,  8, 10,  3,  0, 15,  0,  3,  0,
420     31, 15,  0,  8, 15, 10,  0,  0,  0,  0,  0,
430     31, 10,  0,  8, 15,  0,  0,  0,  0,  0,  0}
440 m_vset(72,v)
450 /*
460 v={
470 /*  AF  OM  WF  SY  SP PMD AMD PMS AMS PAN
480     63, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
490 /*  AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME  E.Organ
500     31, 15,  0,  7,  1, 10,  0,  3,  0,  0,  0,
510     31, 15,  0,  7,  6,  3,  0,  4,  3,  0,  0,
520     31,  0,  0,  7,  0,  0,  0,  2,  7,  0,  0,
530     31,  0,  0,  6,  0,  0,  0,  1,  0,  0,  0}
540 m_vset(73,v)
550 /*
560 v={
570 /*  AF  OM  WF  SY  SP PMD AMD PMS AMS PAN
580     58, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
590 /*  AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME  Strings
600     15,  9,  0,  5,  1, 25,  2,  2,  3,  0,  0,
610     15,  9,  0,  5, 15, 31,  2,  2,  0,  0,  0,
620     15,  0,  0,  5,  0, 25,  1,  2,  0,  0,  0,
630     15,  3,  0,  8,  0,  3,  1,  2,  7,  0,  0}
640 m_vset(74,v)
650 /*
660 v={
670 /*  AF  OM  WF  SY  SP PMD AMD PMS AMS PAN
680     56, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
690 /*  AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME  Hi-Hat Cl
700     31,  0,  0,  2,  0,  0,  0, 15,  0,  3,  0,
710     31,  0,  0,  2,  0,  0,  0, 15,  0,  3,  0,
720     31,  0,  0,  2,  0,  0,  0, 15,  0,  3,  0,
730     31, 18, 13,  9,  7,  5,  0, 15,  0,  3,  0}
740 m_vset(75,v)
750 /*
760 v={
770 /*  AF  OM  WF  SY  SP PMD AMD PMS AMS PAN
780     60, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
790 /*  AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME  Vocal
800     31,  9,  4,  0,  1, 23,  0,  2,  3,  0,  0,
810     31,  6,  4,  6,  2,  5,  0,  2,  3,  0,  0,
820     31,  4,  3,  0,  5, 22,  0,  2,  7,  0,  0,
830     31,  5,  2,  6,  1,  4,  0,  2,  7,  0,  0}
840 m_vset(76,v)
850 /*
860 v={
870 /*  AF  OM  WF  SY  SP PMD AMD PMS AMS PAN
880     40, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
890 /*  AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME  E.Guitar1
900     31,  4,  1,  0,  1, 10,  0,  3,  4,  0,  0,
910     18,  1,  1,  8,  1, 28,  0, 15,  4,  0,  0,
920     31,  4,  1,  0,  1, 23,  0,  7,  7,  0,  0,
930     31, 12,  2,  8,  1,  1,  0,  1,  7,  0,  0}
940 m_vset(77,v)
950 /*
960 v={
970 /*  AF  OM  WF  SY  SP PMD AMD PMS AMS PAN
980     52, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
990 /*  AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME  E.Guitar2
1000    31,  0,  0,  6,  0, 14,  0,  2,  3,  0,  0,
1010    24,  9,  3,  6,  5,  4,  0,  2,  3,  0,  0,
1020    31,  0,  0,  6,  0,  1,  0,  4,  7,  0,  0,
1030    31,  9,  3,  6,  5,  8,  0,  4,  7,  0,  0}
1040 m_vset(78,v)
1050 /*
1060 v={
1070 /*  AF  OM  WF  SY  SP PMD AMD PMS AMS PAN
1080    52, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
1090 /*  AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME  Syn
1100    31,  0,  0,  6,  0, 14,  0,  2,  3,  0,  0,
1110    24,  9,  3,  6,  5,  1,  0,  2,  3,  0,  0,
1120    31,  0,  0,  6,  0,  0,  0,  4,  7,  0,  0,
1130    31,  9,  3,  6,  5,  2,  0,  4,  7,  0,  0}
1140 m_vset(79,v)
1150 /*
1160 v={
1170 /*  AF  OM  WF  SY  SP PMD AMD PMS AMS PAN
1180    52, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
1190 /*  AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AME  Glocken
1200    31,  5,  1,  7,  1, 14,  1,  1,  3,  0,  0,
1210    24,  9,  4,  7,  6,  1,  1,  1,  3,  0,  0,
1220    31,  4,  1,  7,  1,  0,  1,  2,  7,  0,  0,
1230    31,  9,  4,  7,  6,  2,  1,  2,  7,  0,  0}
1240 m_vset(80,v)
1250 /*
1260 goto 1390
1270 /*
1280 /*******************************************************/
1290 /***      S e t   M M L   t o   t r a c k          ***/
1300 /*******************************************************/
1310 func t(tt)
1320  r=0
1330   while o(r)<>255
1340   m_trk2(tt,p(o(r)))
1350   r=r+1
1360  endwhile
1370 endfunc
1380 /*******************************************************/
1390 /***       P l a y   d a t a                        ***/
1400 /*******************************************************/
1410 /*
1420 /******** E . B a s s
1430 p(0)="@70 v14 k0 p3 q8 =0 t170
1440 p(1)="l8o2r2.b16&(b,e)36<ereeeered+d+d+d+>bb<c+d+c+c+cc+
1450 p(2)="c+c+cc64&(c+>a)21bbabbbabaag+aaag+abbabbab4<c+c+ec+
1460 p(3)="f+c+ec+f+c+ec+rv15d+rev14
1470 p(4)="l8o2eef+eeef+e
1480 p(5)="|:aag+a:|bab4bbab<c+rc+>b4ef+g+eef+eeef+eaag+aaag+a
1490 p(6)="bbabbabe4e(g,g+)48eeg+er4v15a4v14|:aag+a:|ag+a4
1500 p(7)="ee<e>be<e>b<e>ee<e>b<ed+e&(e>g)24r4v15a4v14aag+a
1510 p(8)="aag+aag+(a,b)48bb<b4>bbaf+4g+ab4<c+d+>b
1520 p(9)="aag+aag+(a,b)48bbabbbabbbabbab&(b,e)24ef+ef+ef+e4<
1530 p(10)="d+d+dd+d+d+dd+c+c+cc+c+cc+4>bbabbbabaag+aaag+a
1540 p(11)="g+g+gg+g+gg+ef+f+ff+f+gg+a4g+ab4ab4
1550 p(12)="ef+ef+ef+e4<d+d+dd+d+d+dd+c+c+cc+c+cc+4>bbabbbab
1560 p(13)="aag+aaag+abbabbbab<
1570 p(14)="c+c+|:4c+4:|c+c+rv15d+re4>v14e4.eef+e
1580 p(15)="c+c+cc+c+c+(d+,e)48d+c+>b4<c+4(c+,d)48
1590 p(16)="|:3ddc+d:|eedc+|:c+c+cc+:|c+c+c+ed+c+c+c+>
1600 p(17)="|:3g+g+f+g+:|g+f+g+r|:3bbab:|bat200b&(b,e)24t170
1610 p(18)="@73v11l1o4e&eg+g+8&a8g+8e4.r4e1&e2.
1620 p(19)="l8o3e16&(e>b)36|:3bbab:|bab<r4v15c+r4d+4r2rv14>
1630 p(20)="e4ef+eeef+e<|:d+d+dd+:|
1640 p(21)="l8o3|:c+c+cc+:|>|:bbab:||:aag+a:|g+g+f+g+g+f+g+e
1650 p(22)="f+ef+<ec+>baa4g+ab4a+b&(b,e)24|:eef+e:|<|:d+d+dd+:|
1660 p(23)="|:c+c+cc+:|>|:bbab:||:aag+a:|bbabbab4
1670 p(24)="|:eef+e:|<|:d+d+dd+:|cc+cc+ec+4c+>bba+bbg+f+g+
1680 p(25)="|:aag+a:|g+g+f+g+g+f+g+ef+ef+f+f+g+f+a4g+ab4ab4
1690 p(26)="|:eef+e:|<|:d+d+dd+:||:c+c+c:|c+&(c+>g+)24
1700 p(27)="bbabbab&(b,e)24aag+aag+abbbabbab&(b,e)24
1710 p(28)="|:eef+e:|<d+d+dd+d+ed+dc+c+ec+ec+cc+>|:bbg+f+:|
1720 p(29)="|:aag+a:|g+g+f+g+g+f+g+ef+f+ef+f+g+aa4g+abba+b&(b,e
)24
1730 p(30)="|:eef+e:|<|:d+d+dd+:||:c+c+cc+:|>ba+ba+bef+g+
1740 p(31)="aag+aag+abbbabbab&(b,e)24
1750 p(32)="t120@76v13o3g+1&g+1
1760 o={0,1,2,3,4,5,6,7,8,4,5,6,7,9,10,11,12,13,14,5,6,7,9,10,
1770     11,12,13,15,16,17,18,0,19,20,21,22,23,24,25,26,27,
1780     28,29,30,31,28,29,30,31,32,255}
1790 t(1)
1800 /*
1810 /******** V o c a l   e t c .
1820 p(0)="@77 v14 k4 p3 q8 =1 ^10,14,3
1830 p(1)="l8o4v15r1g+4g+f+rarg+&g+2v13g+g+g+&(g+,c)24v15
```

```
 1840 p(2)="g+g+g+f+rarg+4.v13g+g+g+g+g+&(g+,c)24v15
 1850 p(3)="(e,a)12&a4&a16e&e2f+4.g+4.a4g+1&g+1
 1860 p(4)="@76 v15 k4 p3 q8 =0
 1870 p(5)="l8o4rg+g+4g+e4a4ag+4ef+4.rf+f+f+f+g+ag+2a16&g+16f+2
 1880 p(6)="rg+g+g+g+e4g+a4g+e4f+4.rf+f+f+f+g+ae4.r2.
 1890 p(7)="o4r4e2.
 1900 p(8)="l4o4ag+ef+g+8a8g+8e.r @78 l8o4g+&ag+e e2  @76
 1910 p(9)="l4o4ag+ef+<c+.>b8&b2
 1920 p(10)="r1
 1930 p(11)="l8a4.b&b2g+g+g+f+4a4g+4.r4f+g+f+eg+g+g+f+4a4g+2rr2
 1940 p(12)="l8o4a4.e4.<c+4>b4.g+16f+16e4c+eg+4g+f+&f+4.
 1950 p(13)="l8o4@74v12e2f+&f+2
 1960 p(14)="l8o4g+g+g+f+4a4g+2rf+g+f+eg+g+g+l4f+ag+2r8f+g+aa8e.
rf+.g+.a
 1970 p(15)="l2o4g+1&g+r
 1980 p(16)="l2o4g+1&g+.&(g+,d+)24r8
 1990 p(17)="l8o3^28,14,1v15c+32&d2.&d16.^10,14,3=0ef+4f+g4a4c+1
6c+16
 2000 p(18)="=1c+2&c+(c+,d+)36&d+4>(g+<d+)36&(d+,c+)24>g+4.
 2010 p(19)="g+(e,g+)48&g+f+<c+4>b<(f,g+)24&g+f+(b<c+)48>b2<=0
 2020 p(20)="(c+,c)48c>b<e4>b<f+4>b<e4>b<(d+,e)24&(e,d+)24&(d+>d
+)48
 2030 p(21)="@74v12o4e1
 2040 p(22)="&b1 r8a.b2 r.g+g+8g+8f+ag+2l8rf+g+f+e
 2050 p(23)="l8o4g+g+g+f+4a4g+2rr2l4a.e.<c+>b.g+16f+16el8c+eg+4a
f+4.r@74v13o4c+2d+&d+2
 2060 p(24)="l8o4g+g+g+f+4a4g+2rf+g+f+eg+g+g+f+4a4g+2rf+4g+4a4ae
&e2f+4.g+4.a4
 2070 p(25)="l8o4g+g+g+f+4a4g+2rf+g+f+e
 2080 p(26)="l1o4g+&g+
 2090 p(27)="l8o4rg+g+4 g+e4a4a g+4ef+4.rf+f+4f+g+a4g+4.f+4.r4
 2100 p(28)="l8o4rg+g+g+g+e4g+a4g+e4f+4.rf+f+f+f+g+ag+2f+4.r4
 2110 o={0,1,2,3,4,5,6,0,7,4,8,0,7,4,9,10,27,6,0,7,4,8,0,7,4,9,
 2120     11,12,13,4,14,15,28,6,0,7,4,8,0,7,4,9,11,12,13,4,14,
 2130     16,0,17,18,19,20, 21,4,8,21,4,9,22,23,4,24,25,23,4,24,
 2140     25,23,4,24,25,23,4,24,26,255}
 2150 t(2)
 2160 /*
 2170 /******** V o c a l   e c h o   e t c .
 2180 p(0)="@77 v14 k8 p3 q8 =1 ^10,14,3
 2190 p(1)="l8o3v15r1g+4g+f+rarg+&g+2v13g+g+g+&(g+,c)24v15
 2200 p(2)="g+g+g+f+rarg+4.v13g+g+g+g+g+&(g+,c)24v15
 2210 p(3)="(e,a)12&a4&a16e&e2f+4.g+4.a4g+1&g+1
 2220 p(4)="@76 v12 k8 p3 q8 =0 r8
 2230 p(7)="o4r8c+2.
 2240 p(12)="l8o4a4.e4.<c+4>b4.g+16f+16e4c+eg+4g+f+&f+4
 2250 p(13)="l8o3@74v11a2b&b2
 2260 p(16)="l2o4g+1&g+.&(g+,d+)24
 2270 p(17)="l8o2^28,14,1v12c+32&d2.&d16.^10,14,3=0ef+4f+g4a4c+1
6c+16
 2280 p(21)="@74v12o4c+1
 2290 p(23)="l8o4g+g+g+f+4a4g+2rr2l4a.e.<c+>b.g+16f+16el8c+eg+4a
f+4. @74v13o3a2b&b2
 2300 p(24)="l8o4g+g+g+f+4a4g+2rf+g+f+eg+g+g+f+4a4g+2rf+4g+4a4ae
&e2f+4.g+4.a
 2310 p(25)="l8o4v15g+1&g+2"+p(4)+"f+g+f+e
 2320 p(29)="l4o4ag+ef+g+8a8g+8e.r8r1
 2330 o={0,1,2,3,4,5,6,0,7,4,8,0,7,4,9,10,27,6,0,7,4,8,0,7,4,9,
 2340     11,12,13,4,14,15,28,6,0,7,4,8,0,7,4,9,11,12,13,4,14,
 2350     16,0,17,18,19,20, 21,4,29,21,4,9,22,23,4,24,25,23,4,
 2360     24,25,23,4,24,25,23,4,24,26,255}
 2370 t(3)
 2380 /*
 2390 /******** E . G u i t a r   e t c .
 2400 p(0)="@77 v12 k2 p3 q8 =0 ^10,14,3
 2410 p(1)="l8o4r1e4eerered+4d+d+rd+rd+c+4c+c+rc+rc+>bbbbbbr4
 2420 p(2)="aaaaaara bbbbbbrb <v13=1ee1&e4.v15=0rd+rev12
 2430 p(3)="l8o4reerd+erereerg+ara>rb<f+ebf+>b<c+c+c+4>b4>ef+g+
 2440 p(4)="ee<<erd+ere rc+arg+ara r4f+ebf+e>e4.>(g,g+)48<
 2450 p(5)="e>(g,g+)48<er4v14=1a2.=0v10|:8e:|>v12ee<erd+er2v9ee
 2460 p(6)="v12(e,d+)48&(e,d+)48r4v14a2.v12aar4aa(a,b)48
 2470 p(7)="f+f+b4f+b4c+4>g+a<f+4c+d+>b
 2480 p(8)="v13|:7b:|b4bbbbbb&(b,e)24<v12er4er4e4d+d+d+d+rd+d+4
 2490 p(9)="|:4c+:|rc+c+&(c+>f+)24bbbb<erd+>baaaar4aa<er4er4e>b
 2500 p(10)="f+r4f+f+gg+v14a2b&b2v12<er4er4eed+d+rd+r4d+d+
 2510 p(11)="c+r4c+r4c+(c+>f+)24br4.br(b,a)48<c+r4c+r4c+>abr4brb
bf+<
 2520 p(12)="v13=1c+1&c+2v15=0rd+rev12
 2530 p(13)="o3v14c+r2c+4c+c+ed+c+>g+g+<c+4dr2.=1c+d+4r4.=0d4d16
d16
 2540 p(14)="v12|:6c+:|c+4|:4c+:|v14d+c+c+c+>g+rv12|:12g+:|r<
 2550 p(15)="v13=1(a+,b)48b=0>v12bbbv13bv14bv15b v14bbbbb4(b,e)4
8
 2560 p(16)="@79v13l1o4e&eg+l8g+&ag+e&e2e1&e1@74o4d+2e2f+1
 2570 p(17)="l8o4v15rc+r4d+4r2rv13o3e4rer4er4d+r4d+r4d+(d+>g+)24
<
 2580 p(18)="c+r4c+r4c+c+>br4bbrbbaaaar4aa<er4er4eec+r4c+r4.>
 2590 p(19)="v15a2b&b2< v13er4er4eed+r4d+r4d+d+c+r4c+r4c+c+>
 2600 p(20)="br4bbrb&(b,e)24ar4ar4a&(a,d+)24br4br4b&(b,e)24
 2610 p(21)="v13l8o3er4er4e&(e>a)24<d+r4d+r4d+&(d+>g+)24<
 2620 p(22)=p(21)+p(18)+p(19)+p(20)
 2630 o={0,1,2,3,4,5,6,7,3,4,5,6,8,9,10,11,12,3,4,5,6,8,9,10,
 2640     11,13,14,15, 16,0,17,18,19,20,22,22,22, 255}
 2650 t(4)
 2660 /*
 2670 /******** E . G u i t a r   e t c .
 2680 p(0)="@77 v12 k6 p3 q8 =0 ^10,14,3
 2690 p(1)="l8o3r1e4eerered+4d+d+rd+rd+c+4c+c+rc+rc+>bbbbbbr4
 2700 p(2)="aaaaaarabbbbbbrb<v13=1g+g+1&g+4.v15=0rc+64&d+16..rd+
64&e16..v12
 2710 p(3)="l8o3rbbra+brbraar<d+erer2f+r>f+g+g+g+4f+4r4.
 2720 p(4)="l8r4bra+brb<r4erd+erer2..>>b4.r4br4b
 2730 p(5)="r4v14a2.&v13a1r1r1r4v14a1&a2(a,b)48v12
 2740 p(6)="bb<f+4>b<f+4>f+4r4b4r4.
 2750 p(7)="<v13|:7f+:|f+4|:5f+:|f+&(f+>b)24<v12er4er4e4d+d+d+d+
rd+d+4
 2760 p(8)="|:4c+:|rc+c+&(c+>f+)24bbbb<f+rf+f+>aaaar4aa<er4er4ee
>
 2770 p(9)="f+r4f+f+gg+v14a2b&b2v12<er4er4eed+d+rd+r4d+d+
 2780 p(10)="c+r4c+r4c+(c+>f+)24br4.br(b,a)48<er4er4e>abr4brbbb
 2790 p(11)="<v13=1c+1&c+2v15=0rc+64&d+16..rd+64&e16..v12
 2800 p(12)="o2v12c+r2c+4c+c+ed+c+>g+g+<c+4dr2.ar2rd4d16d16
 2810 p(13)="v11|:6c+:|c+4|:4c+:|v13a+g+g+g+>g+rv11|:12g+:|r4
 2820 p(14)="v11bbbbbv12bv13bv14b v13bbbbb4(b,e)48
 2830 p(15)="@79v13l1o4c+&c+el8v10g+4&ag+e2v13c+1&c+1@74>b2b2<d+
1
 2840 p(17)="l8o4v15rb64&<c+16..r4c+64&d+16..&d+r2r@80v13o7e&e1d
+1
 2850 p(18)="l8o7c+1e2@77o4erd+>b@80o7c+1e1@77o4c+r4c+r4.v14c+2d
+&d+2
 2860 p(19)="@80v13o7e2.@77o4e>b@80o7d+2.@77o4d+d+@80o7e2.@77o4c
+c+
 2870 p(20)="@80o7d+2.@77o3b&(b,e)24ar4ar4a&(a,d+)24br4br4b&(b,e
)24
 2880 p(21)="@80o7e2.@77o4e&(e>a)24@80o7d+4d+4e4d+4e2.@77o4c+>g+
 2890 p(22)="br@80o7e4f+4e4c+2.@77o3aa<er4er4e>bf+r2g+rv14a2b8&b
2v13
 2900 p(23)="@80o7e2.@77o4e>b<d+r@80o7|:4g+4:|r2@77o4c+c+>br
 2910 p(24)="@80o7g+4f+4g+4a2.@77o3a&(a,e)24br4br4b&(b,e)24
 2920 p(25)="@80o7g+2.@77o4e&(e>a)24<d+r@80o7|:4g+4:|r2@77o4c+>g
+br
 2930 p(26)="@80o7e4f+4g+4e2.@77o3aa@80o7e2.@77o4e>bf+r2g+rv14a2
b8&b2v13
 2940 p(27)="@80o7g+2.@77o4e>b<d+r@80o7|:4g+4:|r2@77o4c+c+>br
 2950 p(28)="@80o7g+4f+4g+4a2.@77o3a&(a,d)24@80o7f+2.@77o3b&(b,e
)24
 2960 p(29)=p(25)+p(26)+p(27)+p(28)
 2970 o={0,1,2,3,4,5,6,3,4,5,7,8,9,10,11,3,4,5,7,8,9,10,12,13,
 2980     14,15,0,17,18,19,20,21,22,23,24,29,29,            255}
 2990 t(5)
 3000 /*
 3010 /******** E . O r g a n   e t c .
 3020 p(0)="@73 v10 k9 p3 q8 =0 ^10,14,3
 3030 p(1)="l1o4red+c+>babb&b
 3040 p(2)="@77v11l8o3reerd+erereerg+ara v8rrb<f+ebf+>b v11
 3050 p(3)="c+c+4>b4v8ref+g+ee<<erd+ererc+av11cc+rc+v8
 3060 p(4)="r4.f+ebf+>>v11e4.v8r4.<e>gv11e<
 3070 p(5)="r4v13e2.&e1v8r8>ee<erd+er2eer4.
 3080 p(6)="r4v13e2.&e1v8r8f+f+b4f+b4c+4>g+a<f+4c+d+
 3090 p(7)="r4v13e2.&e1>bbbbbbbb4bbbbbb&(b,e)24
 3100 p(8)="l1o4ed+c+>bag+f+2..v11a2b8&b2<v10ed+c+>bab<
 3110 p(9)="v11c+&c+2r2
 3120 p(10)="v11c+&c+ @77v14l8o2ar1r2a4a16a16v12|:6g+:|g+4
 3130 p(11)="|:4g+:|r2v14d+rv12|:12d+:|r4v12|:5f+:|
 3140 p(12)="v13f+v14f+v15f+ v14|:4f+:|f+4(f+>b)48
 3150 p(13)="@79v13l1o2a&a<b@78l8o4g+&ag+e&e2@79l1o2a&a<@74f+2f+
2
 3160 p(14)="@77v13l4o4f+.e.f+l8o3v15rc+r4d+4r2r<@73v10e&e1d+1
 3170 p(15)="v10l1o4c+>bag+f+2..v11a2b8&b2<v10ed+c+>bab<
 3180 p(16)="ed+c+>bag+<e2..>   v11a2b8&b2<v10ed+c+>bab<
 3190 p(17)="|:  ed+c+>bag+f+2..v11a2b8&b2<v10ed+c+>bab< :|  @76
v13o4c+1&c+1
 3200 o={0,1,2,3,4,5,6,2,3,4,5,7,0,8,9,2,3,4,5,7,0,8,
 3210     10,11,12,13,14,15,16,17,255}
 3220 t(6)
 3230 /*
 3240 /******** S t r i n g s   e t c .
 3250 p(0)="@74 v12 k1 p3 q8 =0 ^10,14,3  l1 o4
 3260 p(1)="red+c+d+c+d+c+&c+2@77v15l16o5r8c+64&d+..r8d+64&e..
 3270 p(2)="@77v11l8o4r1r2cc+rc+|:6r1:|v13r4 e2.&e1r1
 3280 p(3)="@78v12l8o4g+ag+e=1e2.=0 @77v13 e2.&e1
 3290 p(4)="l1rr
 3300 p(5)="ed+c+ec+ec+2..v13c+2d+8&d+2v12ed+c+d+c+d+
 3310 p(6)="v13e&e2@77v15l16o5r8c+64&d+..r8d+64&e..
 3320 p(7)="v13e&e |:8r:|
 3330 p(8)="@79v13l1o3e&e@74g+&g+@79e&er@77l4o4d+.c+.d+>r8a8rbr
 3340 p(9)="l1o4r4.e&e8d+c+ec+ef+2..v13e2f+8&f+2v12ed+c+d+ ed+
 3350 p(10)="ed+c+ec+ec+2..              v13e2f+8&f+2v12ed+c+d+c+d+
 3360 p(11)="ed+c+ec+ee2..v13e2f+8&f+2 @79
 3370 p(12)="l8o5|:20v12erec+erv10e16&(e,c+)12r8<:| @76v12o3g+1&
g+1
 3380 o={0,1,2,3,4,2,3,0,4,5,6,2,3,0,4,5,7,8,0,9,10,11,12, 255}
 3390 t(7)
 3400 /*
 3410 /******** D r u m s
 3420 p(90)="@75o1y2,2cy2,2ry2,16c@72d
 3430 p(0)="@72 v15 k0 p3 q8 y15,0 y3,3 =0
 3440 p(1)="l8o1y2,15r32y2,16r16. ddy2,63r32y2,12r16.ddy2,13rd
 3450 p(2)="@72y2,3d4@75y2,16ccy2,2cy2,2cy2,16cc
 3460 p(3)="y2,2ccy2,16ccy2,2cy2,2cy2,16cc
 3470 p(4)="y2,2ccy2,16ccy2,2cy2,2c@71o5y2,16cy2,2c@75o1
 3480 p(5)="@71o5y2,2c4@75o1y2,16ccy2,2cy2,2cy2,16cy2,2c
 3490 p(6)="@71o5y2,2c4@75o1c@72|:y2,62r16:||:y2,62d:|y2,63ry2,6
3d
 3500 p(7)="@72y2,3d4|:5y2,2d4:|y2,2d@71o5y2,16cy2,2ry2,16co1
 3510 p(8)="@75y2,2r4y2,16c4y2,2c4y2,16c4
 3520 p(9)="y2,2c4y2,16cy2,2ccy2,2cy2,16c4
 3530 p(10)="y2,2c4y2,16c4y2,2c4y2,16c@71o5y2,2c@75o1
 3540 p(11)="@71o5ry2,2cy2,16c4|:@72o1y2,64d:|@71o5y2,16c@72o1y2
,62d16y2,63r16@75
 3550 p(12)="@72y2,13dry2,16d4@75c4y2,16c4
 3560 p(13)="y2,2c4y2,16c4@71o5y2,2c&y2,2cy2,16c4@75o1
 3570 p(14)="@72y2,64dy2,64ry2,16ry2,64r|:y2,64d:|y2,16ry2,62r
 3580 p(15)="@72y2,64dy2,64ry2,16ry2,64r|:y2,64d:|y2,16ry2,62r16
y2,63r16
 3590 p(16)="@72y2,64dy2,64ry2,16d4@71o5c4y2,16c4@75o1
 3600 p(17)="@72|:y2,64d:|@71o5y2,16c4@72o1|:y2,64d:|@71o5y2,16c
y2,2c
 3610 p(18)="@71o5r|:y2,16r:|y2,2c4|:3y2,16r:|
 3620 p(19)="@72o1y2,2r|:4y2,16d:|y2,16ry2,16r32y2,16r16.@72d
 3630 p(20)="@72o1y2,3d4@75y2,16c4y2,2c4y2,16c4
 3640 p(21)="@72o1y2,3d4@75y2,16c.c16y2,2cy2,2ry2,16c4
 3650 p(22)="@75y2,2c4y2,16c4"+p(90)
 3660 p(23)="@75y2,2c4y2,16c4y2,2cy2,2ry2,16c4
 3670 p(24)="@71o5y2,2c4@75o1y2,16c4y2,2cy2,2ry2,16c4
 3680 p(25)="@71o5y2,2c4@75o1y2,16c4y2,2cy2,2ry2,16c@71o5y2,2c
 3690 p(26)="@75o1ry2,2ry2,16cy2,2rc4|:4y2,16r16:|
 3700 p(27)="@71o5y2,2c4@75o1y2,16cy2,63r16y2,62r16@72|:y2,64d:|
```

▶そういえば最近「LIVE in '91」にゲームミュージックが少ないですねえ。でも今月の「暴れん坊将軍」のテーマはなかなか面白い。　菊池　正人(17)福島県

```
y2,16r@71o5y2,2co1
 3710 p(28)="ry2,2ry2,16r@71o5y2,2c4y2,2ry2,16r32y2,16r16.@72o1d
 3720 p(29)="@71o5y2,2c4@75o1y2,16c4cy2,2ry2,16c@72d
 3730 p(30)="@72o1y2,3d4@75y2,16c4y2,2cy2,2ry2,16c4
 3740 p(31)="@75o1y2,2cy2,2r@71o5y2,16c|:y2,62r16:|@72o1y2,62dy2
,63r|:y2,64d:|
 3750 p(32)="@75y2,2cy2,16rl16y2,2r|:3y2,16r:|@72y2,16d|:3y2,16r
:|y2,16dy2,16r|:y2,62r:|18
 3760 p(33)="@71o5y2,2c&y2,2c@75o1y2,16c@72d"+p(90)
 3770 p(34)=p(90)+"@75o1y2,2c&y2,2cy2,16c@71o5y2,2c |:y2,2r:|@72
o1y2,16rd|:3"+p(90)+":|"
 3780 p(35)=p(90)+"@72o1y2,16r32y2,16r16.y2,62r16y2,63r16y2,64d4
 3790 p(36)="@72o1|:y2,2d4:|y2,16dy2,16r|:y2,16d:|
 3800 p(37)="@72o1|:5y2,16d:|y2,16r@71o5y2,16c@72o1y2,2d
 3810 p(38)="@72o1d4d4y2,16d4d4
 3820 p(39)="@72o1d@75c@72d4y2,16d4@75cc
 3830 p(40)="|:@72o1d@75c:|@72y2,16d@75c@71o5y2,2c4o1
 3840 p(41)="|:@72o1d@75c:|@72y2,16d@71o5c@75o1y2,2c4
 3850 p(42)="|:@72o1y2,2d:|y2,16r32y2,16r16.ddy2,63ddd
 3860 p(43)="@72o1y2,64rddy2,16rddy2,63dd
 3870 p(44)="@72o1dy2,64d32&y2,16d16.ddy2,16r32y2,16r16.r4.
 3880 p(45)="r4.@72o1y2,3d4@72d@75y2,16cc@72dd@75y2,16cc
 3890 p(46)="@75o1y2,2ccy2,16cc|:y2,2c:|y2,16r|:y2,16r16:|
 3900 p(47)="@71o5y2,2c4@75o1y2,16c4@72dy2,62dy2,16r@71o5y2,2c
 3910 p(48)="ry2,2ry2,16r@71o5y2,2c4y2,2ry2,16r|:y2,16r16:| o1
 3920 p(49)="@71o5y2,2c4@75o1y2,16cc|:y2,2c:|@71o5y2,16c4o1
 3930 p(50)="@72o1dd@71o5y2,16cl16|:y2,16r:|@72o1y2,16dy2,16r|:y
2,64r:|y2,63dy2,63r|:y2,64r:|18
 3940 p(51)="@71o5y2,2c4@72o1y2,16rdy2,16dry2,16r@71o5y2,2c o1
 3950 p(52)="r@72o1y2,2dy2,16r@71o5y2,2c4|:y2,62r16:|y2,63r4 o1
 3960 p(53)="@72o1|:y2,2d:|y2,16rdy2,16d16y2,16ry2,16r16y2,16d16
y2,16r16y2,63r
 3970 p(54)="@72o1y2,3dd@75y2,16c4y2,16c@72dy2,16rd
 3980 p(55)="@72o1y2,16d4y2,16d4y2,16rdy2,16rd
 3990 p(56)="@72o1y2,16d4|:y2,16rd:|@71o5y2,16ry2,2c4o1
 4000 p(57)="@72o1y2,2dy2,16r@71o5y2,2c4y2,16ry2,16r64y2,16r16..
|:y2,16r64y2,16r32.:|
 4010 p(58)="l16@72o1y2,2d|:3y2,16r:| y2,2d|:3y2,16r:|y2,63d|:3y
2,63r:|18y2,64d|:y2,64d16:|
 4020 p(59)="|:5"+p(55)+":|
 4030 p(60)="|:4"+p(55)+":|
 4040 p(61)="@76v13o3y2,3e1&e1
 4050 o={0,1,2,3,3,4,5,6,7,8,8,8,9,8,8,10,11,12,13,14,15,16,13,
 4060     17,18,20,8,8,9,8,8,10,11,12,13,14,15,16,13,21,19,
 4070     24,23,23,23,24,22,25,26,30,23,24,22,24,31,7, 8,8,8,9,
 4080     8,8,10,11,12,13,14,15,16,13,21,19,24,23,23,22,24,22,
 4090     27,28,30,23,24,22,24,29,30,32, 33,34,33,35,36,37,
 4100     38,38,39,40,38,41,42,43,44,45,3, 3,46,2,3,47,48,
 4110     2,3,3,46,49,50,2,3,3,4,2,3,51,52,2,3,3,46,24,53,
 4120     54,59,56,57,54,60,58, 54,59,56,57,54,60,58,61,255}
 4130 t(8)
 4140 m_play():end
 4150 for i=1 to 8
 4160  print "TRK";i;"の残り";m_free(i);"バイト
 4170 next
```

## リスト2　歩いていこう

日本音楽著作権協会(出)許諾第9170457-101号

```
 10 /*
 20 /*      J U N  S K Y  W A L K E R ( S )
 30 /*
 40 /*
 50 /*       A  R u  I  T e  I  K o  O
 60 /*
 70 /*
 80 /*          b y  S . I s h i g a m i
 90 /*
100 m_init()
110 /*
120 dim char v(4,10)
130 /*  BASE
140 v={32,15,2,1,205,25,20,0,0,3,0,
150 30,9,8,6,11,32,2,9,3,0,0,
160 28,6,10,6,11,55,3,0,3,0,0,
170 28,4,3,6,3,32,0,0,3,0,0,
180 28,5,6,6,2,2,5,1,3,0,0}
190 m_vset(70,v)
200 /* E-GUITAR(Oh!X)
210 v={57,15,0,0,0,0,0,0,3,0,
220 31,22,8,6,7,11,2,12,6,0,0,
230 31,6,0,6,3,33,1,3,3,0,0,
240 28,6,0,6,15,32,0,3,4,0,0,
250 31,8,0,8,15,0,0,1,4,0,0}
260 m_vset(71,v)
270 /* HI-HAT(C)
280 v={58,15,0,0,0,0,0,0,1,0,
290 31,0,0,0,12,18,0,15,3,1,0,
300 31,0,0,0,10,35,0,10,3,3,0,
310 31,0,0,0,10,17,0,15,0,3,0,
320 31,18,25,7,14,0,0,1,6,0,0}
330 m_vset(72,v)
340 /* D-GUITAR(BASIC)
350 v={40,15,0,0,0,0,0,0,3,0,
360 31,2,1,0,1,6,0,3,0,0,0,
370 18,3,1,9,1,23,0,15,0,0,0,
380 31,2,1,0,1,28,0,1,3,0,0,
390 31,12,1,8,1,0,0,5,3,0,0}
400 m_vset(73,v)
410 /* E-GUITAR(BASIC)
420 v={16,15,0,0,0,0,0,0,3,0,
430 22,3,4,7,1,36,0,8,3,0,0,
440 24,3,3,7,1,42,0,4,5,0,0,
450 26,7,3,7,1,13,0,4,0,0,0,
460 31,3,2,8,1,5,0,8,0,0,0}
470-m_vset(74,v)
480 /* VOCAL
490 v={61,15,2.1,205,25,20,5,5,3,0,
500 18,1,4,2,1,28,0,1,0,0,0,
510 19,5,6,7,1,11,0,2,0,0,0,
520 12,2,6,7,1,11,2,2,0,0,0,
530 19,8,6,7,1,12,0,2,0,0,0}
540 m_vset(75,v)
550 /* HI-HAT(O)
560 v={58,15,0,0,0,0,0,0,3,0,
570 31,0,0,0,12,18,0,15,3,1,0,
580 31,0,0,0,10,35,0,10,3,3,0,
590 31,8,0,0,10,17,0,15,0,3,0,
600 31,13,25,6,14,0,0,1,6,0,0}
610 m_vset(76,v)
620 /* TOM-TOM
630 v={62,15,0,0,0,0,0,0,3,0,
640 31,4,3,1,5,0,0,15,3,3,0,
650 31,17,13,9,5,8,0,0,7,0,0,
660 31,12,10,9,5,0,0,0,7,2,0,
670 31,12,10,9,5,0,0,0,3,2,0}
680 m_vset(77,v)
690 /* CHORAS
700 v={59,15,2,1,205,25,20,5,5,3,0,
710 31,4,3,5,2,25,1,5,7,0,0,
720 31,5,3,7,2,40,1,3,4,0,0,
730 31,4,3,5,2,25,1,0,2,0,0,
740 31,8,3,5,0,5,1,0,3,0,0}
750 m_vset(78,v)
760 /*  SYMBAL
770 v={59,15,0,0,0,0,0,0,3,0,
780 18,4,3,2,5,22,0,1,3,1,0,
790 19,2,1,1,5,28,0,4,0,2,0,
800 25,11,12,2,3,35,0,11,0,2,0,
810 31,14,11,5,3,0,0,1,7,0,0}
820 m_vset(79,v)
830 /*  DIS
840 v={40,15,0,0,0,0,0,0,3,0,
850 31,4,1,0,1,8,0,3,4,0,0,
860 18,1,1,9,1,22,0,15,4,0,0,
870 31,4,1,0,1,25,0,7,1,0,0,
880 31,12,0,7,1,3,0,1,7,0,0}
890 m_vset(80,v)
900 /*  THUN
910 v={51,15,1,1,0,0,0,0,0,3,0,
920 20,8,4,6,14,44,0,7,-3,0,0,
930 22,7,1,6,10,0,0,0,-3,0,0,
940 22,11,4,8,12,17,0,7,-3,1,0,
950 17,6,0,8,12,9,0,0,-3,0,0}
960 m_vset(81,v)
970 str a[256],b[256],c[256],d[256],e[256],f[256],g[256],h[256
],k[256],l[256],m[256],n[256]
980 str o[256],p[256],q[256],r[256],s[256],t[256],u[256],j[256
],w[256],x[256],y[256],z[256]
990 for i=1 to 8:m_alloc(i,10000):m_assign(i,i):next
1000 /*
1010 /*
1020 a="t170 @70 @v127 o3 q6
1030 b="l2dc+>ba<l8eeeeeeeee4e4>g+a-4.
1040 m_trk(1,a+b)
1050 /*
1060 a=" @79 v15 o4 q2 l4  y3,3
1070 b="|:y2,5rry2,5rr:|@77v14y2,22c8y2,22c8|:6y2,22c8:||:y2,21
c:|y2,21c8y2,5v15@79b4.@72
1080 m_trk(2,a+b)
1090 /*
1100 a="@73v13o4l8 y50,0 y51,56
1110 b="d2c+2>b2a2<eeeeeeeeq7e4e4q8>v13g+v14a-4.
1120 m_trk(3,a+"<"+b+"o5e2"):m_trk(4,a+b+"o4e2")
1130 /*
1140 a="t170@74v12o2l8
1150 b="y52,0l2f+f+dc+l8g+4|:6g+:|q6g+4g+4q8v13g+v14a-4.
1160 m_trk(5,a+b)
1170 /*
1180 b="@71o3y53,20l2dd>cal8e4|:6e:|q6e4e4q8v13dv14e-4.
1190 m_trk(6,a+b)
1200 /*
1210 b="@74v15y54,40o1l2aaf+el8b4|:6b:|q6b4b4q8v13dv14e-4.
1220 m_trk(7,a+b)
1230 /*
1240 b="@80y55,56o2l2dc+>ba<l8e4|:6e:|q6e4e4>q8v13gv14a-4.
1250 m_trk(8,a+b)
1260 goto 4000
1270 /*
1280 func melo()
1290 /*
1300 /*
1310 /*  ｺﾄﾊﾞｼﾞｬ ﾀﾘﾅｸﾃ
1320 /*
1330 a="l8q6@v127|:o2a4aaaaaa:|<|:d4dddddd:|o2e4<eeeeeee4eeeeee
a4aeeedddc+c+c+>bb<ef
1340 m_trk(1,a)
1350 /*
1360 x="l8@77y2,3@72rcy2,22ccy2,23cy2,23cy2,22cc":y="y2,23ccy2,
22ccy2,23cy2,23cy2,22cc"
1370 m_trk(2,x+"|:7"+y+":|")
1380 /*
1390 a="@74o2v12|:e4eeeeee:||:f+4f+f+f+f+f+f+:||:g+4g+g+g+g+g+g
+:||:e4eeeeee:|
1400 m_trk(5,a)
```

▶そろそろ進藤君が聞きたいです。　　田中　明仁(19)大阪府

```
 1410 /*
 1420 a="@71o3v11|:c+4c+c+c+c+c+c+:||:d4dddddd:||:e4eeeeee:||:c+
4c+c+c+c+c+c+:|
 1430 m_trk(6,a)
 1440 /*
 1450 a="@74o1v15|:4a4aaaaaa:||:b4bbbbbb:||:a4aaaaaa:|
 1460 m_trk(7,a)
 1470 /*
 1480 a="@80o2v13|:e4eeeeee:||:d4dddddd:||:4f+4f+f+f+f+f+f+:|
 1490 m_trk(8,a)
 1500 /*
 1510 /*  ｶﾗﾏﾜﾘ ｼﾃｲﾙ
 1520 /*
 1530 a="|:f+4f+f+f+f+f+f+:|b4bbbbbbb4aag+g+f4|:d4dddddd:|>e4<ee
eeeee4>eef+f+g+g+
 1540 m_trk(1,a)
 1550 /*
 1560 a="y2,23ccy2,22ccy2,22r8.y2,22r16y2,22r8y2,22r8
 1570 m_trk(2,x+y+y+y):m_trk(2,x+y+x+a)
 1580 /*
 1590 a="|:c+4c+c+c+c+c+c+:||:d+4d+d+d+d+d+d+:||:f4ffffff:||:g4g
gaggg:|
 1600 m_trk(5,a)
 1610 /*
 1620 a=">|:4a4aaaaaa:|<|:d4dddddd:||:e4eeeeee:|
 1630 m_trk(6,a)
 1640 /*
 1650 a="|:4f+4f+f+f+f+f+f+:||:a4aaaaaa:||:b4bbbbbb:|
 1660 m_trk(7,a)
 1670 /*
 1680 a="|:c+4c+c+c+c+c+c+:||:f+4f+f+f+f+f+f+:||:d4dddddd:||:e4e
eeeee:|
 1690 m_trk(8,a)
 1700 /*
 1710 /*  ﾅｾﾞｶ ｷﾐﾉﾒﾊ
 1720 /*
 1730 a="|:a4aaaaaa:|<|:d4dddddd:||:e4eeeeee:|a4aeeedddc+c+c+>bb
<ef
 1740 m_trk(1,a)
 1750 /*
 1760 m_trk(2,x+y+y+y):m_trk(2,y+y+y+y)
 1770 /*
 1780 a="|:e4eeeeee:||:f+4f+f+f+f+f+f+:||:g+4g+g+g+g+g+g+:||:e4e
eeeee:|
 1790 m_trk(5,a)
 1800 /*
 1810 a="|:c+4c+c+c+c+c+c+:||:d4dddddd:||:e4eeeeee:||:c+4c+c+c+c
+c+c+:|
 1820 m_trk(6,a)
 1830 /*
 1840 a="|:4a4aaaaaa:||:b4bbbbbb:||:a4aaaaaa:|
 1850-m_trk(7,a)
 1860 /*
 1870 a="|:e4eeeeee:||:d4dddddd:||:4e4eeeeee:|
 1880 m_trk(8,a)
 1890 /*
 1900 /*  ｼﾞﾌﾞﾝ ｦ ｼﾝｼﾞﾃ
 1910 /*
 1920 a="|:f+4f+f+f+f+f+f+:|b4bbbbbbbbaag+g+f+4d4dddddddf4ffffff>
e4<eeeeeee4eeeeee
 1930 m_trk(1,a)
 1940 /*
 1950 a="y2,23c4y2,22r8.y2,22r16y2,22r8y2,22r8@77y2,23cc@72
 1960 m_trk(2,x+y+y+y):m_trk(2,x+y+x+a)
 1970 /*
 1980 a="|:c+4c+c+c+c+c+c+:||:f+4f+f+f+f+f+f+:||:f4ffffff:||:g+4
g+g+ag+g+g+:|
 1990 m_trk(5,a)
 2000 /*
 2010 a=">|:a4aaaaaa:|<|:d+4d+d+d+d+d+d+:||:d4dddddd:||:e4eeeeee
:|
 2020 m_trk(6,a)
 2030 /*
 2040 a="|:f+4f+f+f+f+f+f+:||:4a4aaaaaa:||:b4bbbbbb:|
 2050 m_trk(7,a)
 2060 /*
 2070 a="|:c+4c+c+c+c+c+c+:|>|:b4bbbbbb:|<|:d4dddddd:||:e4eeeeee
:|
 2080 m_trk(8,a)
-2090 endfunc
 2100 /*
 2110 /*
 2120 func woo()
 2130 /*
 2140 /*
 2150 /*  Woo- ｱﾙｲﾃｲｺｳ
 2160 /*
 2170 c="o3|:d4ddddddde4eeeeee>a4aaaaaa|1aaaabb<c+c+:||2o3aaggf+f
+ee:||3o3f+4f+f+f+f+ee
 2180 m_trk(1,c)
 2190 /*
 2200 u="18v15@79y2,5r4y2,3>a4<v12y2,23ay2,23ry2,21ar
 2210 j="y2,23ary2,21ary2,23ay2,23ry2,21ar
 2220 w="y2,23ay2,23ry2,21ay2,21ry2,23ay2,23ry2,21ar
 2230 z="y2,23ay2,23ry2,21ay2,21ry2,23ay2,23ry2,21ay2,21r
 2240 m_trk(2,u+j+j+w):m_trk(2,u+j+j+w):m_trk(2,u+j+j+z)
 2250 /*
 2260 str e2[256],f2[256]
 2270 /*
 2280 g="f+4|:6f+:|g+4|:6g+:||:e4eeeeee:|f+4|:6f+:|g+4|:6g+:||:e
4eeeeee:|f+4|:6f+:|g+4|:6g+:|e4|:6e:|c+4|:6c+:|
 2290 m_trk(5,g)
 2300 /*
 2310 k="d4|:6d:|e4|:6e:||:c+4c+c+c+c+c+c+:|d4|:6d:|e4|:6e:||:c+
4c+c+c+c+c+c+:|d4|:6d:|e4|:6e:|c+4|:6c+:|>a4|:6a:|
 2320 m_trk(6,k)
 2330 /*
 2340 m="a4|:6a:|b4|:6b:||:a4aaaaaa:|a4|:6a:|b4|:6b:||:3a4aaaaaa
:|b4|:6b:|a4|:6a:|f+4|:6f+:|
 2350 m_trk(7,m)
 2360 /*
 2370 o="d4|:6d:|e4|:6e:||:e4eeeeee:|d4|:6d:|e4|:6e:||:e4eeeeee:
|d4|:6d:|e4|:6e:|e4|:6e:|c+4|:6c+:|
 2380 m_trk(8,o)
 2390 endfunc
 2400 /*
 2410 /*
 2420 /*
 2430 melo()
 2440 a="@75o514 c+d8e8&e2.eddde8f+8&f+1r2>b<c+8d8&d2.dc+c+c+d8e
8&e1
 2450 m_trk(3,"v14"+a):m_trk(4,"v12r16"+a)
 2460 a="r2>ab8<c+8&c+c+2eed+dd+&d+2reedc+d&d2rde2ef+8e8&e1
 2470 m_trk(3,a):m_trk(4,a)
 2480 a="r2c+d8e8&e1ddde8f+8&f+1r2>b<c+8d8&d2.dc+c+c+8de8&e1
 2490 m_trk(3,a):m_trk(4,a)
 2500 a="r2>ab8<c+8&c+2.eed+dd+&d+2.eedc+d&d2.deeef+8e8&e1
 2510 m_trk(3,a):m_trk(4,a)
 2520 woo()
 2530 d="d4dddddde4>e2.a1&a2.<@11e16&f+16&g+16&a1618
 2540 m_trk(1,d)
 2550 a="y2,3rr@79v12y2,21ary2,23ay2,23ry2,21ay2,23ry2,5rry2,5rr
r4@76v14y2,21c4
 2560 b="@77y2,10r8.y2,22r16y2,22ry2,22ry2,22ry2,22rcc
 2570 m_trk(2,a+x+b)
 2580 str e2[256],f2[256]
 2590 e="@78v15r2df+e1v13@75c+c+c+dc+8>ba8&a2<@78v15r2df+e2.>@75
v13b<c+>b<c+>b<c+8de8&e2
 2600 e2="@75v14r2df+e1v15c+c+c+dc+8>ba8&a2<v14r2df+e2.v15>b<c+>
b<c+>b<c+8de8&e2
 2610 f="@78v15r2df+e1@75v13
 2620 f2="v14r2df+e1v15
 2630 m_trk(3,"@v127"+e2+f2):m_trk(4,"v13"+e+f)
 2640 a="c+c+c+dc+8>ba8&a2 r<dd8dd8edc+>ba1r1v14
 2650 b="c+c+c+dc+8>ba8&a2 r<dd8dd8edc+>ba1r1v12
 2660 m_trk(3,"@v127"+a+"v14"):m_trk(4,"v13"+b+"v12")
 2670 h="f+4|:6f+:|g+4g+2.e1<<@73e8d8c+8>b8a8g+8f+8e818@74>>
 2680 m_trk(5,h)
 2690 l="<d4|:6d:|e4e2.c+1<c+8>b8a8g+8f+8e8d8c+818>
 2700 m_trk(6,l)
 2710 n="a4|:6a:|b4b2.a1<<@71v11a8g+8f+8e8d8c+8>b8a818@74v15>>
 2720 m_trk(7,n)
 2730 p="d4|:6d:|e4e2.e1@73<e8&d8&c+8&>&b8&a8&g+8&f+8&e818v11@80
>
 2740 m_trk(8,p)
 2750 /*
 2760 a="o3 @v127 |:7a4aa&aaa4:|<eeddc+&d>bbd4dddddd>a4aaaaaae4e
ef+f+g+g+a4aaaaaa<
 2770 b="d4dddddd>a4<c+c+ddd+e>e4<eeeda16&g+16&f+16&e16dc+>ba&aa
<e4
 2780 m_trk(1,a+b)
 2790 /*
 2800 a="@72o4y2,5rcy2,22ccccy2,22cc":b="y2,23cy2,23cy2,22ccccy2
,22cc"
 2810 q="y2,23ccy2,22ccccy2,22cc":r="18@77y2,23c.@72y2,22r16|:y2
,22r.y2,22r16:||:y2,22r:|
 2820 s="@77y2,22cy2,22cy2,22cy2,5c4y2,23r16@76v13y2,22c16&c4v14
 2830 t="@72y2,23ccy2,5ccy2,23cy2,23cy2,5cc
 2840 m_trk(2,a+b+q+b):m_trk(2,a+b+q+r):m_trk(2,x+y+x+t):m_trk(2
,x+y+x+s)
 2850 /*
 2860 a="@74o314 d8ea8&adc+8ea8&ac+>b8<ea8&a>ba8<c+8e8a8&a2d8ea8
&adc+8ea8&ac+>b8<ea8&a8a8>b<@11
 2870 m_trk(3,a+"e2e16&d16&c+16&c16&>b16&a+16&a16&g+1614")
 2880 b="@74o314d16ea8&adc+8ea8&ac+>b8<ea8&a>ba8<c+8e8a8&a2d8ea8
&adc+8ea8&ac+>b8<ea8&a8a8>b<@11
 2890 m_trk(4,b+"c+2c+16&>b16&a+16&a16&g+16&g16&f+16&f1614")
 2900 a="d.e8&edc+.d8&dc+>b.<c+8&c+>ba.b8&b<c+d.e8&edc+.d8&dc+>b
.<c+8&c+dd8c+8>b8a8&a2
 2910 m_trk(3,"o3"+a):m_trk(4,"o2"+a)
 2920 /*
 2930 a="|:7r1:|<<<v11e8d8c+8>b8a8g+8f+8e818@74>>
 2940 b="o218 f+4rrrrf+4e4rrrre4g+4g+g+4g+g+4e4ee4ee4f+4f+f+4f+f
+4e4ee4ee4g+4g+g+4g+g+4eeee&e2
 2950 m_trk(5,a+b)
 2960 a="|:7r1:|v12<<c+8>b8a8g+8f+8e8d8c+818v12@71>
 2970 b="o318 d4rrrrd4c+4rrrrc+4e4ee4ee4c+4c+c+4c+c+4d4dd4dd4c+4
c+c+4c+c+4e4ee4ee4c+c+c+c+&c+2
 2980 m_trk(6,a+b)
 2990 a="|:7r1:|<<<@71v11a8g+8f+8e8d8c+8>b8a818@74v15>>
 3000 b="o118 a4aaaaa4a4eeeea4b4bb4bb4|:3a4aa4aa4:|b4bb4bb4aaaa&
a2
 3010 m_trk(7,a+b)
 3020 a="|:7r1:|v11<<e8&d8&c+8&>&b8&a8&g+8&f+8&e818v11@80>
 3030 b="o218 d4ddddd4e4>aaaa<e4|:e4ee4ee4:|d4dd4dd4|:e4ee4ee4:|
eeee&e2
 3040 m_trk(8,a+b)
 3050 /*
 3060 melo()
 3070 a="@75o514 r2c+d8e8&e2.eddde8f+8&f+1r2>b<c+8d8&dd2dc+c+c+8
de8&e1
 3080 b="r2>ab8<c+8&c+2.eed+dd+&d+2r4eedc+d&d2v15
 3090 m_trk(3,"v14"+a+b+"r4deeef+8e8&e1v14"):m_trk(4,"v12r16"+a+
b+"r8.@78deeef+8e8&e1v12r16")
 3100 a="@75r2c+d8e8&ee2eddd8ef+8&f+1r2>b<c+8d8&d2.dc+c+4d8e8&
e1
 3110 b="r2>ab8<c+8&c+2.eed+dd+&d+2.eedc+d&d2.deeef+8e8&e1
 3120 m_trk(3,a+b):m_trk(4,a+b)
 3130 /*
 3140 woo()
 3150 a=">|:b4bbbbbb:||:3ee<ee&e>e<e4:|>eeeeeeee
 3160 m_trk(1,a)
 3170 /*
 3180 a="y2,5r@72v14c@79v12y2,21a@72v14c@79v12y2,23a@72v14y2,23c
```



▶今日，システムソフトの「ボンバーマン」ゲーム大会へ行ってきました。でも１回戦で負けてしまい，やっぱり高校生にはかなわないと思いました。　梅本　康治(28)大阪府

```
@79v12y2,21a@72v14c
 3190 b="@79v12y2,23a@72v14c@79v12y2,21a@72v14c@79v12y2,23a@72v1
4y2,23c@79v12y2,21a@72v14c
 3200 m_trk(2,a+b)
 3210 a="@79v14y2,5r&ry2,21ay2,21aay2,23ry2,21a4
 3220 b="y2,5r4y2,20r.y2,20r16|:4y2,20r:|
 3230 m_trk(2,a+a+a+b)
 3240 /*
 3250 e="@78v15r2df+e1v13@75c+c+c+dc+8>ba8&a2<@78v15r2df+e2.>@75
v13b<c+>b<c+>b<c+8de8&e2
 3260 e2="@75v14r2df+e1v15c+c+c+dc+8>ba8&a2<v14r2df+e2.>v15b<c+>
b<c+>b<c+8de8&e2
 3270 f="@78v15r2df+e1@75v13
 3280 f2="v14r2df+e1v15
 3290 m_trk(3,"@v127"+e2+f2):m_trk(4,"v13"+e+f)
 3300 a="c+c+c+c+c+8>ba8&a2<rdd8dd8dddde2.@78v15f+e2.p1ee2.@75@v
127ag+.f+8&f+e
 3310 b="c+c+c+c+c+8>ba8&a2<rdd8dd8dddde2.@75v14f+e2.p1v15ee2.@7
5ag+.@v127f+8&f+e
 3320 m_trk(3,a):m_trk(4,b)
 3330 /*
 3340 a="|:f+4f+f+f+f+f+f+:||:3g+4g+g+&g+2:|g+1
 3350 m_trk(5,a)
 3360 /*
 3370 a="<|:d4dddddd:||:3e4ee&e2:|e1
 3380 m_trk(6,a)
 3390 /*
 3400 a="|:a4aaaaaa:||:3b4bb&b2:|b1
 3410 m_trk(7,a)
 3420 /*
 3430 a="|:d4dddddd:||:3e4ee&e2:|e1
 3440 m_trk(8,a)
 3450 /*
 3460 woo()
 3470 /*
 3480 a="df+e1c+c+c+dc+8>ba8&a2<r2df+e2.>b<c+>b<c+>b<c+8de8&e2
 3490 b="r2df+e1
 3500 m_trk(3,"r2@78@v127o5"+a+b):m_trk(4,"r4r8r16@75@v127o5"+a+
b)
 3510 a="c+c+c+dc+8>ba8&a2 r<dd8dd8edc+>ba1r1@75v14
 3520 b="c+c+c+dc+8>ba8&a2 r<dd8dd8edc+>ba1r1v14
 3530 m_trk(3,a):m_trk(4,b)
 3540 d="d4dddddde4>e2.a1&a2.<@11e16&f+16&g+16&a1618
 3550 m_trk(1,d)
 3560 a="y2,3rr@79v12y2,21ary2,23ay2,23ry2,21ay2,23ry2,5rry2,5rr
r4@76v14y2,21c4
 3570 b="@77y2,10r8.y2,22r16y2,22ry2,22ry2,22ry2,22rcc
 3580 m_trk(2,a+x+b)
 3590 h="f+4|:6f+:|g+4g+2.e1<<@73e8d8c+8>b8a8g+8f+8e818@74>>
 3600 m_trk(5,h)
 3610 l="<d4|:6d:|e4e2.c+1<c+8>b8a8g+8f+8e8d8c+818>
 3620 m_trk(6,l)
 3630 n="a4|:6a:|b4b2.a1<<@71v11a8g+8f+8e8d8c+8>b8a818@74v15>>
 3640 m_trk(7,n)
 3650 p="d4|:6d:|e4e2.e1@73<e8&d8&c+8&>&b8&a8&g+8&f+8&e818@80>
 3660 m_trk(8,p)
 3670 /*
 3680 /*   last
 3690 /*
 3700 a="|:6a4aa&aaa4:|<eeed&ddc+4c+>a&bba4a4d4dddddd>a4aaaa<de>
e4f+f+ggg+g+
 3710 b="a4aaaaaa<d4dddddd>a4aaaa<de|:3>e4<eeede4:|>e4eef+f+g+g+
a1&a2a4r4r2
 3720 m_trk(1,a+b)
 3730 /*
 3740 a="@72o4y2,5rcy2,22ccccy2,22cc":b="y2,23cy2,23cy2,22ccccy2
,22cc"
 3750 q="y2,23ccy2,22ccccy2,22cc":r="18@77y2,23c.@72y2,22r16|:y2
,22r.y2,22r16:||:y2,22r:|
 3760 s="@77y2,22cy2,22cy2,22cy2,5c4y2,23r16@76v13y2,22c16&c4v14
 3770 t="@72y2,23ccy2,5ccy2,23cy2,23cy2,5cc
 3780 m_trk(2,a+b+q+b):m_trk(2,a+b+q+r):m_trk(2,x+y+x+t)
 3790 a="@72y2,23ccy2,22ccy2,23cy2,23ry2,22cy2,22c16y2,22c16
 3800 b="@77>y2,5eey2,22eey2,23eey2,22e<c16>e16<
 3810 q="@76y2,5r4y2,22c4@77c4@76y2,22c4 y2,5r4c4c4c4.c4. y2,5r4
r4r2
 3820 m_trk(2,x+a+b+b):m_trk(2,b+q)
 3830 /*
 3840 a="@74o314 d8ea8&adc+8ea8&ac+>b8<ea8&a>ba<e8a8&a8a8>a18<da
edaedac+aec+aec+4>b<ae>b<ae>b4
 3850 m_trk(3,a+"a2a16&g+16&g16&f+16&f16&e16&d16&c1614")
 3860 b="@74o314 d8ea8&adc+8ea8&ac+>b8<ea8&a>ba<18c+e&ee8>a4 <da
edaedac+aec+aec+4>b<ae>b<ae>b4
 3870 m_trk(4,b+"a2a16&g+16&g16&f+16&f16&e16&d16&c1614")
 3880 a="d.e8&edc+.d8&dc+>b.<c+8&c+>ba.b8&b8<c+.a.g+8&g+8f+.18ee
rd&dc+4.>b4bbbbb4<c+4c+c+c+c+c+4d4dddd4e4eeeeeee1&e4>@11e16&d16
&c+16&>b16a16a8.&a8a8r2
 3890 m_trk(3,"o3"+a):m_trk(4,"o2"+a)
 3900 /*
 3910 a="|:7r1:|<<v11e8d8c+8>b8a8g+8f+8e818@74>
 3920 b="o218 f+4rrrrf+4e4rrrre4g+4g+g+4g+g+4e4ee4ee4f+4f+f+4f+f
+4e4ee4ee4>|:3b4bb4bb4:|b4bbbbbb<c+1@11c+16&c16&>b16&a16&g+16&g1
6&f+16&f16<c+16c+8.c+4r2
 3930 m_trk(5,a+b)
 3940 a="|:7r1:|v12<c+8>b8a8g+8f+8e8d8c+818v12@71
 3950 b="o318 d4rrrrd4c+4rrrrc+4e4ee4ee4c+4c+c+4c+c+4d4dd4dd4c+4
c+c+4c+c+4>|:3g+4g+g+4g+g+4:|g+4|:6g+:|a1@11a16&g+16&g16&f+16&f1
6&e16&d16&c+16a16a8.a4r2
 3960 m_trk(6,a+b)
 3970 a="|:7r1:|<<@71v11a8g+8f+8e8d8c+8>b8a818@74v15>
 3980 b="o118 a4aaaaa4a4eeeea4b4bb4bb4|:3a4aa4aa4:||:3e4ee4ee4:|
e4eeeeeee1@11e16&d16&c+16&c16&>b16&a16&g+16&g16<e16e8.e4r2
 3990 m_trk(7,a+b)
 4000 a="|:7r1:|v11<e8&d8&c+8&>&b8&a8&g+8&f+8&e818v11@80
 4010 b="o218 d4ddddd4e4>aaaa<e4|:e4ee4ee4:|d4dd4dd4e4ee4ee4>|:3
b4bb4bb4:|b4bbbbba1@11a16&g+16&g16&f+16&f16&e16&d16&c+16a16a8.a
4r2
 4020 m_trk(8,a+b)
 4030 /*
 4040 /*
 4050 /*   PLAY
 4060 /*
 4070 /*
 4080 m_play()
 4090 end
```

# (善)のゲームミュージックでバビンチョ

はぁーい，おボンジュールざんす。暑い季節がやってきましたね。突然ですが，今月からこの善バビはLIVEのページでやることになりました。ゲームのページを見てガックリした方，なくなったわけではありません。移動しただけです。というわけで，これからもよろしく。まずはお知らせまで。

さて，去る5月12日秋葉原のラジオ会館で「ボンバーマン大会」が行われました。当日私は38度の熱を出してしまいましたが，「見知らぬボンバー戦士と戦ってみたい」という誘惑に負けて病床を抜け出し，参加してしまいました。参加者は約30名程度。「ドクロモード」で2本勝負，トーナメント式に競技が進められました。私の結果は惜しくも2位。熱のせいでまったく無口，しかも冷や汗ダラダラの顔色真っ青。かなり危ない奴と思われてたみたいでした，トホホ。

**●ナムコビデオゲームグラフィティ　VoL.7**

**CD:VICL-8004　ビクター音産　2,800円(税込)**

最新作の「スティールガンナー」や異色作の「ピストル大名の冒険」「球界道中記」，光線銃ゲーム「コズモギャングズ」などのオリジナルサウンドとアレンジバージョン5曲という盛沢山かつ奇妙な内容。

また，「ピストル大名～」アレンジバージョンと銘打って収録されているのが「火縄丸数え歌」。これはなんとあの「タイムボカン」のヴォーカルの山本正之が歌っているのです。山本ファンは要チェックだ！

さて，その他のアレンジバージョンはちょっとね……。その昔の「妖怪道中記」や「ロンパーズ」のときみたいなパワーとスマートさが欲しいな。

・システムⅡってFM8＋PCM24声なんだってさ。MIDI顔負けだね。

お勧め度　　7

**●FORMULA/SST CD:PCCB-00059**

**ポニーキャニオン　　2枚組　3,200円(税込)**

システム32の「ラッドモビール」をはじめ「GPライダー」「R360」などの体感ゲームサウンド満載の2枚組アルバム。DISC1のアレンジバージョン集がなんといっても最高。最近のゲームミュージックは，オリジナルサウンド自体が（音源などの進歩により）ある程度完成された形態を取っているため，どうもアレンジバージョンにしたところで「だからどうした」的なものが多いんだよね。ところが，今回の「FORMULA」のは，あたかもアレンジバージョンを作るために作曲されたかのように，とても各楽器がバランスよく奏でられていて，カシオペアとかT-SQUAREみたいな「プロ」の味がする。曲そのものも聴きやすいし……（もっとも5曲目はカシオペアの野呂一生氏のアレンジだったりするんですけどね）。

一方，DISC2のオリジナルサウンドは，いい意味でも悪い意味でもいつものセガセガしい出来上がりです……。

・ほんと，DISC1はゲームファン以外にもお勧めできると思う。

お勧め度　　9

＊

今月は，2本だけなんですよ。ごめんなさい。そのかわり，といっちゃあなんですが，このあと番外編として，あの「ユーフォリー」の作曲者との対談記事が続きます。なんだかもう，なんでもアリになってきたな。

さあ，そこの立ち読みのボクも，定期購読のアナタも寄ってらっしゃい見てらっしゃい!

▶電脳倶楽部6月号の「What is going on around here?」は4月号のSHIFT BREAKでいっていた付録ディスクで入れようと思ってポシャッてしまった曲ではないのだろうか。

中田　健一(20)東京都

善バビ番外編

# 対談・GMコンポーザー

第1回　システムサコム・ミュージッククリエーター　**斎藤　学氏**

（善）バビ番外編としてまたまた不定期で，最前線で活躍中のゲームミュージック・コンポーザーのお話をうかがう「対談・GMコンポーザー」のコーナーを始めることになりました。第1回のお客様としてあのX1の名作「ユーフォリー」や人気アドベンチャーゲーム「闇の血族」など，数々の名曲を生み出している斎藤学氏をお迎えすることができました。

＊

**西川善司(以下善)**：こんにちは，どうも遅れてスミマセン（30分の遅刻をしたオオタワケ者です，私は）。さっそくですが，やはり古くからのOh!X（Oh!Mz）の読者としてはサコムの斎藤さんというと「ユーフォリー」という印象が強いと思うのですが，あの曲はいつごろ作曲されたものなのですか。

**斎藤学氏(以下斎)**：高校２年くらいのときにサコムでのアルバイト時代に作曲したものですから17くらいのときですか。

**善**：ええっ。それはすごいですね。斎藤さんの曲は覚えやすいたいへん美しいメロディの曲が多いといわれますが。

**斎**：はい，いつもそれは心掛けていますね。

**善**：一番初めに勉強された楽器というのは？

**斎**：ピアノです。ピアノは４歳のときから入社する直前までレッスンに通ってました。

**善**：作曲はどのように行われているのですか？

**斎**：うーん。通勤電車の身動きひとつできない状況のなかで，やることがないせいもありますが，考えて，会社・自宅に着いたら，まぁ，それを弾くなり打ち込むなりして曲に仕上げていきますね。キーボードは社にあるあのちっちゃいヤマハのDX100（４オペFM音源シンセ）を使うだけでそう大それた機材は特には使いませんね。

**善**：「ユーフォリー」や「ヴァルナ」(PC-8801のARPG)「プロヴィデンス」（同じくPC-8801のARPG）などはたいへん曲数が多いですけど，あれはどのくらいの期間でお作りになったんですか？

**斎**：１日コンスタントに２曲。

**善**：!!

**斎**：PC-8801の場合はノーマル音源用，サウンドボードIIと２種類のデータを作らなければならないので１日に打ち込みは４曲分やってました。はっきりいって大変でしたよ。

**善**：……すごいですね，それは。話は変わりますが，MIDI対応に伴って斎藤さんの曲調が変化したと感じたのですが。

**斎**：ま，あの調子（「ユーフォリー」や「ヴァルナ」のような曲調）のを何百曲作ったって自分の進歩がないと判断したのも理由のひとつですが，MIDI対応の第１回作品は「38万キロの虚空」というゲームだったし，ゲームの雰囲気自体も変わってましたから。

**善**：ああ，そういえばそうですね(笑)。で，サコムのMIDI対応はオリジナルの音色を使われてますよね。

**斎**：ええ，初期の頃は内蔵音色しか使っていませんが……。「ジェミニウィング」の発売が延びましたよね，あのとき時間ができたので（笑）LA音源（MT-32の音源のこと）の勉強をしまして，ま，いまはオリジナルの音色のストックから使うといった感じです。

**善**：そうそう，FM-TOWNSのゲームで「エボリューション」てありましたよね，あの１面の曲が好きなんですよ，あのわざとの不協和音っぽいコード進行の。あれって名曲ですよね。私なんかOh!FMからFM-TOWNSも持ってないのにソフトCDを借りて曲だけ聴いていたんです（笑）。

**斎**：ありがとうございます。ま，ちょっと前衛的なものを狙ったんですよ，あれは。あれはDX7，MT-32，U-110などを使ってまして，これはMT以外のは全部個人のものでスタッフが持ち寄ったんですよ（笑）。

**善**：そういえば「エボリューション」には「ユーフォリー」の水路のシーンの曲のアレンジバージョンが入ってましたね。

**斎**：よくご存じですね。あれは「ユーフォリー」の中で一番気に入っている曲なんです。

**善**：ソロとか入ってたりして！

＊　　＊　　＊

**善**：開発環境はいいんですか。

**斎**：ええ，最近16ビットが主流になってから楽になりましたよ。「ユーフォリー」や「ヴァルナ」の頃は１曲2Kバイト（2048バイト）以内といわれてましたからね。

**善**：に，2Kバイト??

**斎**：ええ，もっと長い曲とかソロとか入ってたりとかベースの凝ったやつとか，オブリガート（助奏）があったりする曲とかもあったんですが，容量の関係で縮めざるをえなかったり，やむなくボツにしたり……。

**善**：2Kバイトというのは凄い世界ですね。

**斎**：某ソフトハウスさんは1Kバイトだったそうですよ。それで私は「斎藤さんのところは2Kバイト，２倍もある，なんでもできるじゃないですか」なんていわれたこともありました。

**善&斎**：ぶはははは!!（大爆笑）

**善**：い，1Kバイト!!!　まぁ，X1やPC-8801は基本的にメインメモリは64Kバイトですからねぇ。

**斎**：ま，いまは，X68000なんかは100Kバイト単位でもらえますからね。よくプログラマに「PCM使うから300Kバイトもらえる？」と聞くと「それでもいいけど200Kバイトのほうがいい」……と，100Kバイト単位の世界ですよ。

**善**：んー……。

**斎**：あの頃は１バイトに泣くなんてケースもありましたよ。打ち込んでコンパイルしたら曲データが2049バイトとか。１バイト削れったって削れるものじゃありませんからね(笑)。ま，当時は作曲よりもメモリとの戦いが苦しかったです……（しみじみ）。

**善**：んーんー……。

**斎**：ま，そういうのをユーザーに感じさせてはいけないわけですからね，大変ですよ。

**善**：（さすが）……。

＊　　＊　　＊

**善**：好きなアーティストはありますか。

**斎**：そうですね。新しいところでは筋肉少女帯ですかね。クラシック，ジャズなどあらゆるジャンルに精通していないとああいう曲はできないと思います。まあ，そう自分で判断しているわけですが，よく聴きますね。

**善**：んー……(ちょっと意外)。……ゲームミュージックでは？

**斎**：コンパイルさんのが好きですね。「ザナック」とか「ガンヘッド」とか。ゲーム自体も好きです。(笑)

**善**：ほかには？

**斎**：アーケードメーカーよりもパソコン関係のほうをよく聴きますね。アーケードのが別に嫌いとかいうわけじゃなくて。アルシスの「ナイトアームズ」やズームの「ジェノサイド」，古代祐三氏の「スキーム」とかはかなり好きなほうですよ。

**善**：サコムのCDというのは……？

**斎**：いまのところ「38万キロの虚空」のみ発売中です。

**善**：以前，通販のみのカセットテープをお作りになったそうですが，あれはまだ在庫はあるのですか。

**斎**：いえ，いま，社に１本もないんです。増版の予定もいまのところありませんね。

**善**：そうですか。アルシスみたいな「ベストセレクション」のようなCDが出ればいいのに。

＊

と，このほか，サコム製のMIDIボード発売のとき，「MIDIボードを付けたのに音が鳴らない」という質問が１日何十件もあり，楽器がないと音が鳴らないことを説明するのに追われて仕事が手につかなかったときの話とか，時間があればMT-32以外のMIDI音源にも対応していきたい，しかしほかのMIDI音源に対応する際にはやはりその音源をフルに使いこなせるまで勉強をしたいというようなこともおっしゃっていました。

というわけで，斎藤さんは天性の音楽的才能を持ちながらも向上心に燃える努力家，という印象を私は強く受けました。これからも頑張ってください。みなさんもシステムサコムと斎藤学氏を応援してあげてくださいネ。

それでは，第２回の実現を祈りつつ……（少しでも反響があれば……)。

参考資料

## 斎藤学氏が担当したGMの数々

（ ）内は斎藤氏の作曲当時の年齢，※は他の人との共作ということを表しています。また「メルヘンヴェールミュージックライブラリ」は作曲は別の人で，打ち込み，アレンジ作業を斎藤氏が担当しています。また◎は斎藤氏のお勧め，☆は私（善）のお勧めです。

(17)　メルヘンヴェールミュージックライブラリ(PC-8801)
(17)☆ユーフォリー(X1)
(17)　ファイヤーロック
(17)※DOME(PC-9801/X68000他)
(17)　シャティ(PC-9801/X68000他)
(18)　ソフトでハードな物語(PC-9801/X68000他)
(18)◎幽霊君(MSX2)（いまでもタケルで買えるそうです）
(18)☆プロヴィデンス(PC-8801)
(18)☆ヴァルナ(PC-8801)
(19)☆エボリューション(FM TOWNS)
(19)　ソフトでハードな物語２(PC-9801/X68000他)
(20)　38万キロの虚空(PC-9801/X68000他)
(20)※闇の血族・上巻/下巻(PC-9801/X68000他)
(20)　ジェミニウィング(X68000)
(20)　アトミックロボキッド(X68000)

●編集部へのメッセージ

今月号の特集について

いちばん良かった記事

興味のなかった記事

これから載せてほしい記事内容

本誌以外にお読みのパソコン雑誌

推薦する市販ソフト

ソフト名：

推薦理由：

最近買って後悔したものは何ですか

あなたの愛機は(所有機種に○印をつけてください)　ない

X1(マニアタイプ,C,D,F,G,twin) X1 turbo(model 10,20,30,40,II,III,Z,ZII,ZIII)

MZ-(80K/C, 1200, 700, 1500, 80B, 2000, 2200, 2500, 2861)

X68000(初代,ACE,PRO,PRO II,EXPERT,EXPERT II,SUPER,XVI, HD) その他

FD(　基)　TAPE　QD　HD(　MB)　MO　プリンタ(　)

年齢　歳　パソコン歴　年　男・女　プレゼントNo.

切り取り線

通常払込料金
加入者負担

# 払込通知票

| 口座番号 | 東京 | 1 | | 29307 | 金額 | |
|---|---|---|---|---|---|---|
| 加入者名 | ソフトバンク株式会社 | | | | 料金 | 払込み / 特殊 円 |
| 払込人住所氏名 | ※ （郵便番号　　　　） | | | | 備考 | |
| | | | | | 受付局日付印 | |

各票の※印欄は、払込人において記載してください。

記載事項を訂正した場合は、その箇所に訂正印を押してください。

この払込通知票は、機械で使用しますので、下部の欄を汚さないよう特に御注意ください。また、本票を折り曲げたりしないでください。（郵政省）

切り取らないで郵便局にお出しください。

通常払込料金
加入者負担

# 払込票

| 口座番号 | 東京 | 1 | | 29307 |
|---|---|---|---|---|
| 加入者名 | ソフトバンク株式会社 | | | |
| 金額 | | | | |
| 払込人住所氏名 | ※ | | | |
| 備考 | | 受付局日付印 | | |

# 振替用紙

←点線から、きれいに切り取ってご使用ねがいます。

切り取り線

【定期購読のご案内】

定期購読のお申し込みを頂きありがとうございます。定期購読料金は以下の通りですが、この申込書の弊社到着〆切は全誌、発売日の10日前です。これを過ぎますと次号からの発送に繰上げさせていただきます。尚、この申込書は郵便局で払い込まれてから弊社到着まで2週間ほどかかります。また、ご希望の雑誌のお届けは書店発売日より遅れる場合がありますのでご了承下さい。

〈定期購読料金〉

| 誌名 | 期間・料金 | 発売日 |
|---|---|---|
| Oh! PC | 年間 12,320円 | 毎月1・15日発売 |
| | 6ヶ月 6,160円 | |
| THE COMPUTER | 年間 7,440円 | 毎月18日発売 |
| Oh! X | 年間 7,200円 | |
| Oh! FM | 年間 6,720円 | |
| C MAGAZINE | 年間11,760円 | |
| パソコンマガジン | 年間 7,680円 | |
| Oh! Dyna | 年間 6,960円 | |
| 月刊情報処理試験 | 年間 8,160円 | 毎月8日発売 |
| | 6ヶ月4,080円 | |

(ご注意)

バックナンバーからの購読申し込みは出来ません。お近くの書店でご注文ください。

切り取り線

切り取らないで郵便局にお出しください。

この欄は、加入者あての通信にお使いください。

## 送り先

| | フリガナ | 性別 | 年齢 | ご職業 |
|---|---|---|---|---|
| お名前 | | 男・女 | | |

| | フリガナ |
|---|---|
| ご住所 | 〒 |

| | ご自宅 | 勤務先 |
|---|---|---|
| お電話 | | |

## 定期購読申込書

| | | | | | | |
|---|---|---|---|---|---|---|
| THE COMPUTER | 新規 | 継続 TC NO. | | 月刊情報処理試験 | 新規 | 継続 JS NO. |
| Oh!/PC | 新規 | 継続 PC NO. | | | | |
| Oh!/X | 新規 | 継続 X NO. | | | | |
| Oh!/FM | 新規 | 継続 FM NO. | | | | |
| パソコンマガジン | 新規 | 継続 PM NO. | | | | |
| C MAGAZINE | 新規 | 継続 CM NO. | | | | |
| Oh!/Dyna | 新規 | 継続 D NO. | | | | |
| | | | | | | |

通信欄

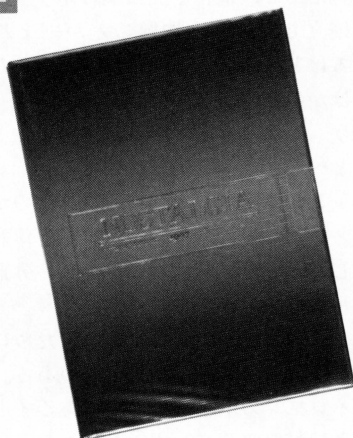
この払込通知票は、機械で使用しますので、下部の欄を汚さないよう特に御注意ください。また、本票を折り曲げたりしないでください。 (郵政省)

# 愛読者プレゼント

## プレゼントの応募方法

とじ込みのアンケートはがきの該当項目をすべてご記入のうえ，希望するプレゼント番号をはがき右下のスペースにひとつ記入してお申し込みください。締め切りは1991年7月18日の到着分までとします。当選者の発表は1991年9月号で行います。

### 1

タケル
☎03(3839)1013

5名

## ノスタルジア

X68000用　5"2HD版4枚組　11,800円(税別)

時代は1907年，豪華客船の乗っ取り事件を解決していくアドベンチャー。画面も時代に合わせて美しいセピア調だ。

### 2

NCS
☎03(3486)6588

3名

## シグナトリー

X68000用　5"2HD版5枚組

12,000円(税別)

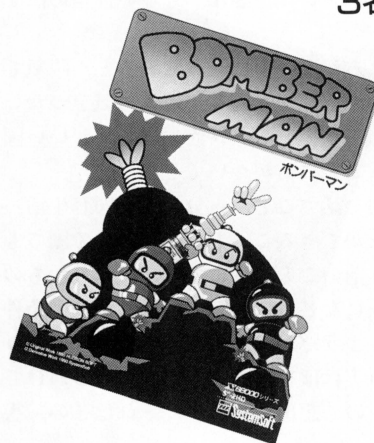
ニューヨークを拠点に繰り広げられる近未来SFアドベンチャーゲーム。5章からなる大作だ。タイムトラベラー気分でどうぞ。

### 3

システムソフト
☎092(752)5262

## ボンバーマン

X68000用　5"2HD版　7,800円(税別)

3名

その昔発売された爆弾男のリメイク版。昔のゲームらしく，ルールも簡単，友達とワイワイいいながらやりたいゲームです。

### 4

アイレム
☎06(535)4880

## イメージファイトテレカ

5名

イメージファイトに同梱されている非売品テレカ。デザインはシンプルだが，キラキラ光るキレイなテレカだ。

### 5

ソフトバンク
☎03(5488)1360

## 激変する半導体産業

5名

本誌の連載でもお馴染みの高原秀己氏が書いた本。高原氏のサイン入りでプレゼントしちゃいます。

## 5月号プレゼント当選者

1哭きの竜（長野県）黒沢剛（岡山県）難波英樹　2ラプラスの魔（茨城県）若月功（岡山県）石原忠　3ノスタルジアCD（青森県）葛西良憲（東京都）秋元乾太郎（千葉県）金子哲也（神奈川県）二宮秀一（静岡県）三須薫（香川県）西池陽一（愛媛県）本田和正（長崎県）山辺由紀子　4コミックガンガンテレカ（神奈川県）由岐中康司（静岡県）青島一高（京都府）福知健　5MENKURI（東京都）志水浩（愛知県）柴田尚宏（鹿児島県）福留敏和（敬称略）

以上の方々が当選されました。おめでとうございます。商品は順次発送いたしますが，入荷状況などにより遅れる場合もあります。また，公正取引委員会の告示により，このプレゼントに当選された方は，この号の他の懸賞には当選できない場合がありますのでご了承ください。

X-OVER・NIGHT
(クロスオーバーナイト)

## [第13話]

# 内側から見たアメリカ

TAKAHRA HIDEKI 高原 秀己

何回か前に，アメリカという国について好きじゃない旨の原稿を書いたが，そのアメリカに2週間弱ほど出かける機会を得た。テキサス，西海岸のハイテク事情を取材に行って以来1年半ぶりで，2度目になる。今回はロッキーマウンテン沿いに西部を回ったあと，ちょっとニューヨークに寄ってくるという，アメリカの過去と現在を探訪するような旅行だった。いうまでもなくエレクトロニクス分野とはまったく関係のない取材だった。

外から見るアメリカは，鼻持ちならない傲慢さをぷんぷんとさせ，常にトップリーダーとして周りから奉られていないと気がすまないという，とんでもない性格の国である。

ところがその中に入ってみると，陽気で開放的で，しかも人々が他人に気をつかいながら生きているという，居心地がいい国であることに気がついてしまう。

英語を話すこと，自分たちと同じ生活パターンができることという，どこの国でも当たり前のことは別にすると，自分たちのルールになじめることという最低限の条件はあるようだが，その条件さえクリアできれば（実はぼくはクリアできているわけではないので想像なのだが)，それほどの障害はない。

もちろん，これは地方ごとに違うだろうし，自分がどんな人種であるかによっても異なった感想を抱くだろう。また，実際に長期にわたって生活していくと，いろいろと複雑なこともあるだろう。それは映画「ゴッドファーザー」を見るまでもなくあきらかであり，現実に根深い問題として横たわっている。

だが，「～だから絶対に受け入れられない」とか，「～だから，～する資格がない」という，根本的差別思想自体は存在しない点には注意する必要がある。

これは，いまのアメリカ人自体が欧州から勝手に住みつき，原住民を追い払って成り立った国であるということがバックボーンとして大きいのだろう。つまり，自分の何代か前の祖先が外来者であったのだから，外来者に対して基本的にオープンな姿勢を保っている，ということだ。

もちろん，そうはいっても人間であるから，既得権益という発想は当然ながらある。「自分はそうだったかもしれないが，これからは違う」という考え方が頭をもたげてきたとき，本音と建て前との戦いになってしまうことはいうまでもない。

しかし，くどいようだがやはり基本的にはオープンな国なのだ。これは日本の典型的村社会や英国貴族社会はもちろんのこと，アメリカと類似の歴史的経緯をたどったオーストラリアや南アフリカ共和国などとはまったく違う点である。

◇

西部を旅していると，つくづく大きな国であることを痛感してしまう。四方八方，すべてに地平線が見えるフラットな大地。その風景は何時間も車で走りつづけても変わらないのである。

こうした大きな国であり，しかもそこで種々雑多な大量の人々が集まって生きてきたからこそ，アメリカではコンピュータが誕生し，これを道具とする「システム」という考え方が，飛躍的に発達したのだろうな，と（漠然とだが）納得した。

たとえば，圧倒的に広大なエリアを対象にして，なんらかの物を大量に売っていこうとすると，さまざまな基盤が必要になる。物を作って届けるデリバリーシステム。どれだけ売れたから，さらにどれだけを作るかという仕組み。さらに，自分の目の届かないほど遠くの場所でも同じ仕事をしてもらうフランチャイズ方式。

物流や交通を見ると，さらに念が入っていることに驚いてしまう。航空業界ではアメリカは文句なしのナンバーワンなのだが，あれだけ広い国で，しかも乗り継ぎがある客を前提としてちゃんと運行してきたというのは感心する。鉄道は欧州のほうが優れているし，日本も相当なレベルにあるのだが，飛行機の場合はちゃんと空を飛んでいって目的地に到着するという次元の異なる作業があるし，何よりも預けた荷物が乗り継いだ飛行機にちゃんと積載されているという違いがある。鉄道にはこの作業がない。船だと作業時間が違う。

こうしたことを着実に進めてこれた原動力が，「システム」という考え方である。レンタカーしかり，チェーンレストランしかりである。

ただし，情報や交通網のネットワークが確立し，時間と距離の差が縮まれば縮まるほど，本来アメリカが得意だった点は，ほかのどの国でもできるようになってしまう。そうなると，勝負は緻密に仕上げることへとグレードアップする。

この緻密さでナンバーワンなのが，日本ということだ。精密産業，半導体産業を制したことを見れば，いかにこの分野で強いかはあきらかだ。

この次元での勝負になったために，日本が経済競争で優位に立ちつつある。セブンイレブンの本社を提携先であるイトーヨーカ堂が買収してしまったことなどは，その典型的事例だろう。

◇

ところで，ニューヨークの一角で驚いた出来事を最後にひとつ紹介しよう。パソコンとかテレビとかの電子機器を売っていた店のショーウィンドウをなにげなくながめていたところ，なかなかしゃれた白い小型ラップトップパソコン（「98NOTE SN」くらいの大きさね）を発見。名前を見て仰天してしまったんだな，これが。

「MZ-200/250」。

うひゃあ，こんなところで生きていたのかMZ。しばし，ショーウィンドウの前で感慨にふけってしまった。オーストラリアでウルトラマンG（グレート）を見つけた人もこんな感じだったんだろうか。

ちなみにスペックは10MHzのi8088と新品にしては古典的なCPU。3.5インチドライブ1基で，上位機種のMZ-250には20MBのハードディスクも入っていた。ごくふつうのIBM互換パソコンだ。

しかし，ただただニューヨークの一角にMZパソコンが存在していることに，ぼくはなんともいえぬ感情でいっぱいになり，価格を店員に聞くこともなく，その場を去ってしまったのである。

# 急にNeXTを使いはじめる

## 使われなかったNeXT

大学の僕の机のすぐ横に，真っ黒な直方体のボディを持った計算機が置いてあります。もういうまでもないのかもしれませんが，NeXTです。国内でのNeXTの販売が開始されてから，1年と8カ月ほどになろうとしていますが，この部屋に初めてやってきて真っ黒なサイコロを見つける学生のうちの何割かは，いまもなお，「あっ，NeXTだ！」などと小声でもらします。

すでに，何十台ものNeXTを学生の教育用に並べるような国立大学（神戸大学）も出現しはじめています。また，昨年末から市場に出た新しいNeXTも出足は好調のようです。ですから，学生が驚くのは単にめずらしいというだけでなく，僕が本誌など（文献1）でアピールしているような魅力が，一般的な認識として定着してきたことからではないかと思ったりします。

学校の教育用にNeXTを使う，つまり最初に使うマシンがNeXTのようなきわめて使い心地のいいマシンであるということは，自動車教習所で最新式のオートマチック車を用意するようなことだといわれるかもしれません。特に理系で，しかも計算機自体を作り出すことを専攻とする学生の場合には問題がないとはいえないでしょう。まあ，そういう問題は横に置くとして，とにかく僕は市場に出回っているワークステーションの中ではこのNeXTがあいかわらずダントツなマシンであると思っています。

ところがです。残念ながら僕自身NeXTをあまり使っていないという歴然とした事実もあるのです。自分ながらもったいない話だと思います。これにはひとつの小さな理由とひとつの大きな理由があります。小さな理由のほうはプライベートな話です。つまり，このマシンが僕自身，あるいは研究室の所有物ではなく，いつまでも使えるという保証がないので，あまり深入りできないということです。

大きな理由については，もうあまり大声でいうまでもないでしょうが，スピードが遅すぎるということです。このことについてはもういやというほどいわれ続けてきました。が，もう大丈夫です。なぜなら，ここにあるNeXTのCPUは68030ですが，新しいNeXTでは68040を使っており，4倍ほど速くなっているのですから。

しかも，スティーブ・ジョブズはNeXTのソフトウェアについては15年の寿命があるとまでいい切ってその思い入れの強さを示していますが，CPUに対する思い入れはほとんどないといっていいでしょう。速ければ何でも採用するといっていますから，これからもぐんぐん速くなることでしょう。

## 新しいOSが送られてくる

先日，キヤノンから新しいNeXT用OSが送られてきました。ついに，日本語への対応がなされたのです。もちろん，すぐにインストールすることにしました。まだ，試作版なのですが，試作版だからこそ，すぐに試してみたくなるといったほうが正解でしょう。

このNeXTのOSは以前はバージョン1が載っていましたが，新しいNeXTシリーズからはバージョン2になっており，当然送られてきたものもバージョン2でした。僕にとってはOSのバージョンが上がり，さらに日本語化もなされたというわけです。

インストールの手作業自体はワンタッチで終わります。ただ，2,3時間待っている必要があります。MO（光磁気ドライブ）からのインストールなので，こんなに時間がかかるのもしかたないといえばしかたないでしょう。ちなみにこのMOは新シリーズではオプションとなってしまいました。

インストールがワンタッチであるといっても，このNeXTはイーサネットで学内LANにつながれているので，そちらの設定もしなおす必要があります。こちらに関してはちょっとした手作業が必要になります。もし，複数のNeXTをつないだLANならば，NeXT独自のネットワークサービスのシステム（NetInfoというもので，いわゆるYPより高級なもの）を利用して，もっと簡単できめの細かい管理ができるのです。

いちおう，新しいOSがインストールできたので，電源を入れて立ち上げてみます。立ち上げてみてまず気づくのは，以前のNeXTではブラックホール，SX-WINDOWではクリーナー，普通のMacintoshではごみ箱，そして僕の研究室のMacintoshでは洋式便器である（ファイルが捨てられていると本当に臭そうなのです），ファイルを捨てるためのアイコンが，リサイクルを表すマーク（図1）に変わっているということです。名前もそのまま，リサイクラ，と呼ぶようです。

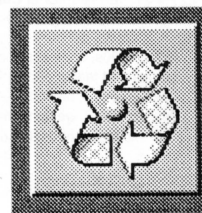
これはエコロジー運動の最近の高まりを示している変化であると考えられましょう。僕自身もこのマークを知らなかったのですが，手元に偶然このマークが押されている封筒があり，ああそうなのかと気がつきました。ちなみに，この封筒はアメリカからフロッピーが送られてきたときに使われていたもので，要するに再利用できますよという意味なのです。

たしかに，ファイルをここに持ってくるということは，文字どおりファイルを捨てることなのですが，復活もできますよという意味も小さくないわけで，案外リサイクラというのもぴったりな感じもします。ただし，ファイルを持っていったときのアイコンの動きとしては，前のブラックホールのほうがダイナミックであり，見せていたなと思います。

## さらに極まったデスクトップ

郵便ポストのアイコンを見ると中に郵便が入っています。さっそく，そのアイコンをダブルクリックすると，スティーブ・ジョブズからメイルが来ています。メイルの中には彼の顔写真が入っています（図2）。最後のほうには唇マークも入っています。これがNeXTお得意のリップサービス（ボイスメール）です。

唇マークをダブルクリックすると，サウンドプレイヤーが呼び出され，プレイボタンを押すと再生が始まります。内容はおなじみの「インターパーソナルコンピューティング」（文献1）の宣伝です。新しいユーザーを作ると自動的にこのメイルが送られてくるようです。

図1　リサイクラ

アイコンやデザインなど細かいところで変化が見られますが，いちばん大きな変化はファイルビューア（ディレクトリの中身や木構造が見られるウィンドウ）の上部に「シェルフ」と呼ばれるサブウィンドウが追加されたことです（図3）。

シェルフには任意のディレクトリ（やシェルフ）を登録できます。これにより，特定ディレクトリへの移動やディレクトリ間のファイルのコピーがずいぶんと楽になりました。以前はいちいち目的のディレクトリを開いておかなければなりませんでした。MacintoshやSX-WINDOWでも画面に余裕のある場合には，このようなサブウィンドウを設定できるといいですね。

そのほか，NeXT OS 2.0では次のような変更がなされました（文献2）。

**1） Objective-C++**

NeXT OS 1.0ではObjective-Cが公用語として採用されていましたが，C++がしだいに普及しつつある現在では，少し異色でした。そこでウルトラCといいましょうか，両方が混在したコードを処理できるコンパイラ，Objective-C++を開発したのです。ただし，システムで用意しているクラスライブラリはObjective-Cにかぎられているなどという制限があります。

**2） プリント機能の高速化**

NeXT OSの実体はMachと呼ばれる分散OS（UNIX互換を保っている）です。その特徴であるスレッド（プロセスより細かい処理の単位）を利用することにより，プリント中の待ち時間を減少させました。

**3） インタフェイスビルダの改良**

ユーザーインタフェイス部を自動作成できるインタフェイスビルダが改良され，第三者（サードパーティ）の作った画面上のオブジェクト（部品）を自分の部品として使えるようになりました。

## 優雅だから目立つ無神経さ

少しいじっているうちに，「日本語版というけれど，いったいどこが？」という重大な疑問がふくらんできました。テスト版ですから，ろくなドキュメントももちろんありません。日本語対応していそうなソフト（たとえばJDRAW）などを立ち上げても，日本語の入力も表示もできません。

この問題はすぐに解決されました。要するに，ホームディレクトリなどに特定の設定やファイルなどが設定されていないと日本語化の恩恵は受けることができないというわけです（まあ，当然といえば当然でしたが）。

ユーザーの登録をNeXT独自の自動的なツールで行えば，日本語化の設定も自動的にできてしまうのです。ところが，僕はユーザー定義のファイル（/etc/passwd）などをネットワーク経由で別のマシンから取ってきて，それをniloadコマンドで変換するという方法をとったため，日本語化がなされなかったのです。

日本語化の設定がなされると，メニューも日本語になります。笑えるのが「ハイド」です。何でしょう，この「ハイド」というのは？　EditやInfoはそれぞれ編集，ガイドなどと日本語に直しておきながら，Hide（画面からウインドウを一時的に消す）をそのまま「ハイド」とは。特に，NeXTのように細かいところまでの異常な凝りようを見せつけるマシンでは，このような無神経さが特別に目立ってしまいます。

図2　ジョブズからのメイル

**Welcome to the NeXT world.**

**1980s: Personal Computing**
*Mission: Improve individual productivity and creativity.*

The mission of Personal Computing was to improve individual productivity and creativity. And this mission was, for the most part, successfully accomplished during the 1980s. But that's not enough anymore...

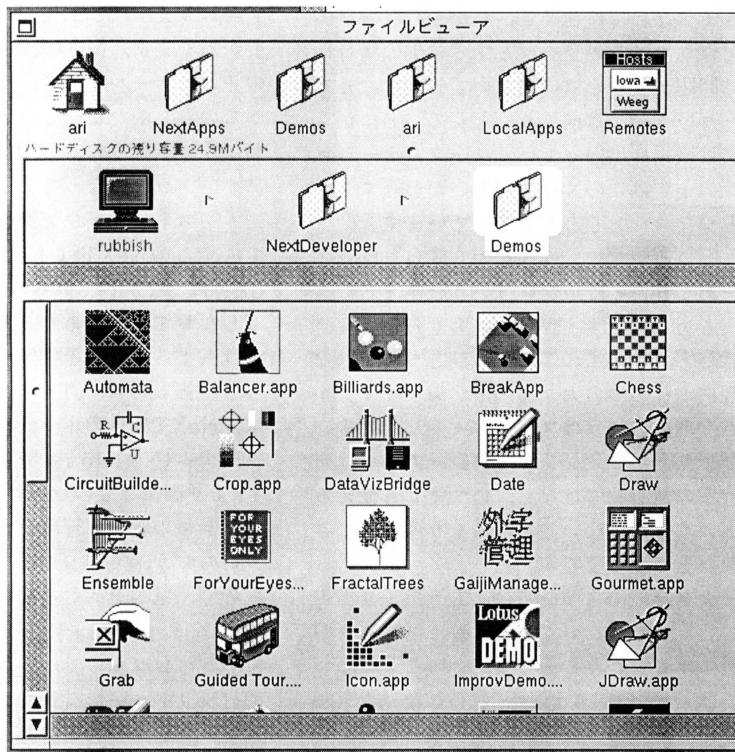
**1990s: Interpersonal Computing**
*Mission: Improve group productivity and collaboration.*

In the 1990s, competitive advantage will come from improving the productivity of groups and teams of people. We need to transform Personal Computing into Interpersonal Computing—and your NeXT computer was designed to do exactly this. At NeXT, we think that Interpersonal Computing will be the most important and exciting contribution of desktop computing in the 1990s.

*Please double-click here:*

図3　シェルフ（上側の家のアイコンのあるところ）

Arita Takaya 有田 隆也

第50回 知能機械概論——お茶目な計算機たち——

# 急にNeXTを使いはじめる

ファイルビューアのメニューもこのような日本語化がなされ，ファイル名に日本語が使用可能となり，日本語対応ソフト（Edit,JDRAW,Terminalなど）で日本語の入力が可能になりました。いちおう，日本語の問題もクリアできたというわけです。

日本語入力FEPは悪くない出来でしょう。キー割り当てなども柔軟にできるようです。NeXTは画面表示もレーザープリンタと同じような方式を取っています（ディスプレイポストスクリプト)。漢字もポストスクリプトデータのようで，エディタで大きさを変えてもきれいなままでした。

## これこそオブジェクト指向

NeXTのインタフェイスビルダは有名ですが，僕自身は表面的にいじったことはあっても実際に何かを作ったりしたことは一度もありませんでした。これがその名のとおり，基本的にはユーザーインタフェイス部分の作成を容易にしてくれるにすぎないと思っていたからです。

たしかに，ソフトハウスが売りものにするアプリケーションソフトを開発する場合には，ユーザーインタフェイスにかける作業量は全作業量の6割にも上るかもしれません。ならば，このインタフェイスビルダはずいぶんとありがたいものになるでしょう。しかし，僕は売りものになるソフトを書いているわけではないのです。

とはいうものの，せっかくNeXTをいじりはじめたのですから，へりくつをいっていてもしかたありません。文献3にサンプルプログラムとして載っている電卓を作ってみることにしました。

インタフェイスビルダを使ったプログラミング作業は次のようなステップで行われます。

**1) 画面のレイアウト**

すでに用意されているメニュー，ボタン，つまみなどをレイアウトして，作成するプログラムの画面デザインを行います。これらメニュー，ボタンなどひとつずつをオブジェクトと呼びます（画面に表示されないオブジェクト＝プログラムもあります）

**2) コネクションの定義**

オブジェクト間の関係をマウスで直接指し示すことによって，そのオブジェクトにイベントが起こったときにどのような機能を実行するか設定します。たとえば，あるボタンを押したとき，ある手続き（これはプログラミングをする必要がある）を実行させるには，ボタンから手続きを記述するオブジェクトまでマウスをドラッグすると，画面上で実際にリンクされます。さらに手続きオブジェクトの中の手続き名（正確には，対応するメソッドのセレクタ名）を指定すればいいのです

**3) ソースファイルの生成と追加**

1),2)によって定義された情報を実際のObjective-Cのソースコードに落とします。ただし，手続きの中身などは空白になっていますので，当然自分でプログラムする必要があります。メインルーチンも自動的に作成されます。Macintoshでおなじみのイベントループ（マウスなどで起動されるとそれに対応した処理を行う）です

このあとはコンパイルすれば終了です。インタフェイスビルダはユーザーインタフェイス部を自動的に作ってくれるだけといえるかもしれませんが，意義深いのはオブジェクト指向に忠実であるということです。オブジェクト指向はSmalltalkでもおなじみの概念ですが，この概念は次の3つで性格づけることができます。

**1) メッセージパッシング**

オブジェクト間でメッセージを交換することによって計算は進む（手続きに引数を渡すなどという概念はない）

**2) インヘリタス（継承）**

オブジェクト（というよりはクラス）は木構造をしており，より根元で定義された手続き（メソッド）などを利用できる（人のプログラムを自由に利用できる）

**3) カプセル化**

データと手続きをひとまとまりにしてカプセル化（オブジェクト化）し，情報を隠蔽することができる（勝手にデータが変更されたり，データや手続き名が衝突したりする心配がない）

このような概念は，現実世界のモデル化として，比較的自然であると考えられています。したがって，インタフェイスのような目に見える部分への応用にきわめて向いているといえましょう。Macintoshなどでもすでに利用されていましたが，その場合はそれこそ目に見える部分だけに関してのオブジェクト指向化でしたが，NeXTの場合はそれを実現している言語のレベルまでオブジェクト指向化がなされているという点で，本質的な違いがあるといえます。

## 終わりに

ばたばたとNeXTを使いはじめた顚末記を書いてきました。使いこなすのはまだまだこれからといえましょう。しかし，オブジェクト指向をインタフェイスの部品に適用し，それをオブジェクト指向型言語で実現するというアプローチはきわめて自然で美しいと実感しました。

なんとか動くようになった電卓を図4に示します。これ以上シンプルな電卓はないというくらいそっけない電卓です。でも，すぐにでもデザインや機能を自由に変更/拡張できる可能性を持っているところが，ひと味違う電卓なのです。

ところで，大学関係者のためのNeXT懇談会というのが開かれ，ちょいと参加することになりました。なにかとっておきの話があったら，またご報告することにしましょう。

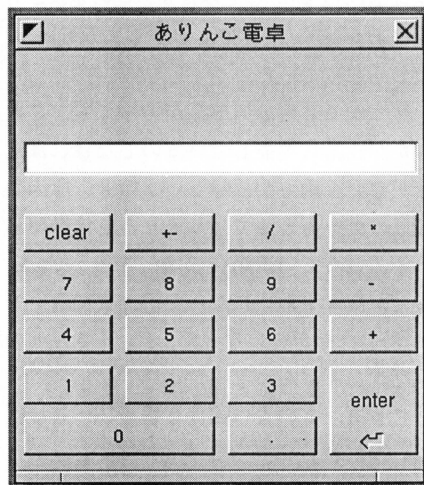

**参考文献**

1) 有田隆也：知能機械概論「ジョブズはやっぱり天才だ！」，Oh!X 1991年1月号，pp.172-173

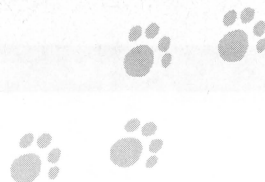
2) NeXTユーザー会：「NeXT OS 2.0 格段に使いやすくなった新OS」，スーパーアスキー 1991年6月号，pp.66-75

3) NeXTユーザーズガイド，キヤノン


図4 どシンプルな電卓

第60回

# 猫とコンピュータ

# TK-復活の午後

Takazawa Kyoko
高沢 恭子

**長年連れ添ってきたTK-80が脳死状態になってしまい，ちょっぴり元気をなくしていたキョウコさん。しかし，TK復活を目指しあれやこれややっているうちに，意外なところに落とし穴が……。**

お彼岸を過ぎた，ほんのりあたたかい日曜日の午後3時，イシイさんはTK-80の往診におとずれた。黒のブルゾンに黒のジーンズ，工具をつめこんだバッグも黒のイシイさんが玄関にあらわれると，ホンニャアは予想どおり，すぐ逃げた。

## 合同治療

脳死状態のTK-80をなんとかよみがえらせようという努力は，その後もつづいていた。病巣を本格的につきとめるには，健康な部品と1つひとつ交換してみるのがいちばんと，イシイさん宅に保管されていた，コバヤシ先生のコンポBS（TK-80 BSがカバーにおさめられた形のもの）が，ちょっと前にわが家に運ばれてきていた。

ところが，見かけは元気そうなコンポBSだったが，フタを開いてみると，あちらこちらゆがんだり，ハンダづけが取れたりしていた。長いことイシイさん宅で眠ったままになっていて，じっさいには動かすこともなかったから，こちらの健康状態もあやしいものだ。

夫はハンダをつけなおし，形を整えて，コンポBSを動かしてみたが，案の定，こちらはわが家のTK-80よりさらに反応がおかしくなっていた。

せっかく2台分の部品があっても，どちらかが正常に機能しているのでなければ意味がない。両方が故障というのも，原因さがしのゲームとしてはおもしろさ倍増だけれど，遊んでいる場合ではなさそうだ。なんだか変なぐあいになったが，まずはコンポBSの修繕が先になって，イシイさんとの電話がひんぱんになった。

「じゃあ，この青い線は2番めにつなげばいいんだね」

きのうも水晶発振器の線が切れているのがあらたにみつかって，またイシイさんとのやりとりになった。ちゃんと確認しないとあとの作業にひびくことになる。ハードマニア筆頭のイシイさんはおおいにたよりになるものの，電話の話はなかなかもどかしい。おまけにそれが1日おきくらいになってきたから，イシイさんもたまらなくなった。

「やっぱり，じかに見たほうが話が早いからネ」と，とうとう本業のデザインのしごとを放り出して，7つ道具をかつぎ，片道2時間あまりの道のりをやってきたのだ。

2人の男性が出会ったとたん，元気の火花が家じゅうに飛び交った。なんといっても機械大好き同士。あまりにも年期モノのマシンの故障なのだけれど，原因をさぐり修復させることは，マシンをひとつつくるような楽しさがあるのだろう。意見を交わしながらのしごとは効果も大きいし，ひとりのときとは活気もちがう。思いがけず開かれたハードの研修会のようで，私まで元気になる。

とはいっても，マシンルームの床は解体された2つのマシンの部品でいっぱいになり，壁ぎわの緑色のラックの中では，わが家のTK用のモニタテレビの画面がかげろうのようにユラユラと動いている。

これらのマシンは，TK-80のボードとBSのボードから構成されている。2人はこの2種のボードを，双方のマシンでかわるがわる交換して組み合わせ，テストをくりかえしていた。4通りの実験はどれもエラーの続出で，復活の見込みは明るくないのに，なぜか声がはずんでいる。

## 2ミリ角の決め手

イシイさん所有のTK-80のボードとは，以前に交換のテストが済んでいて，これもよい結果は出ていなかった。

「なんとなく，ウチのTKがいちばん正常に近い感じだなぁ」と夫が言うと，

「そう，ふだん動いていないのはダメだね。人間もそうね」と言いながら，イシイさんは元の形にもどったわが家のTK-80 BSのキーを叩いた。

「9FFE」と，BASICを走らせるための命令を入力する。「OK」が表示される。なんだか動き出したらしい。

「あ，動き出したかな，もう一度やってみよう」。夫が交代する。

「CFFE」と，ふだんやっているようにつづいて入力する。

「あれ？」イシイさんはなんだかヘンだなという顔をして，たしかめるようにBSボードに目をやった。

「ここサ，いつもこうなの？」

メモリの領域を指定するディップスイッチは，白いプラスチックで2ミリ幅くらいのものが8つならんでいる。スイッチを倒したときに見える小さな正方形の面が，1つだけ目立って白い。ほかの7つと汚れぐあいがちがって真新しいのだ。これはどう考えても，反転させて間もないと思われる。

「あーっ」「はっハッハッ！」

2人は同時にこんな声をあげた。

BSのメモリ配置は各自で設定し，それをディップスイッチの切り替えで指定するようになっている。イシイさんと夫では設定の場所が異なっているので，イシイさんが自分のマシンのつもりで命令を入力すると，当然，わが家のマシンとは反応がちがってくる。命令によって「OK」となったり，「パラメータアヤマリ」になったりする。

それでなんとなくディップスイッチに目をやったとき，ちかごろ指定を変えた形跡のある1つを発見したのだ。

なにか力が加わったのか，意図してやったのを忘れてしまったのか，ともかくスイッチは設定したつもりとは逆の方向になっていた。夫は以前のとおり命令を入力していたので，あて先の番地と思っていたところには何もなかったというわけだ。これではどんな健康な部品と交換してみても，結果はエラーにしかならないはずだ。

電話と宅配便を往復させて，たくさんの実験をしたり，古いマシンのマニュアルを頭でたどったり，長い努力だった。最後は名医の登場であっと驚く結末の物語になったが，ついでに2台まとめてTK-80のオーバーホールができたのは拾いものだ。

一件落着のあと，トオルもまじえての4人の食事はこのうえなくゆかいだった。まるで難事件を解決したあとの特捜班の控室のように，捜査の筋書きをたどりながら，またあらたな事件を想定してみたり，それがよもやま話に変わっていったり。

話の合間に私が，食卓のそばの小さなワゴンに乗ったノートパソコンを見せて，

「ほら，おかしいでしょ，ノートパソコンにもう1台キーボードをつないでるの」と，イシイさんがあきれて笑い出すのを期待しながら言うと，

「そういうパソコンって中がいじれないでしょ，ボク興味ないの」と，しんみり真顔でつぶやいて，オン・ザ・ロックのグラスを干した。自分で設計し，組み立て，思ったとおりに動かせる機械がほんとうに好きなイシイさんなのだ。

## そしてプリントが残った

高校生活のスタートで，トオルの環境はたくさんの新しいことが始まった。徒歩1分から電車とバスの通学へ。給食からお弁当へ。外見では紺のブレザーから黒のつめ衿へ。これはボタンのない伝統のスタイルだそうだが，なんだか「中国の服みたいだな」と，いっしょに通学が決まったドイ君が言う。

もっと重大な変化は，学校が示す勉強の方針や学習内容で，その量も質も当然中学校の比ではない。表面さり気なくふるまっているものの，トオルは環境を受け入れるために緊張の日々を送っている。

トオルの卒業より少し遅れて，PTA委員としての私の任務も，4月末で終了になった。かろうじてつとめていた「校外補導委員会」のしごとだ。

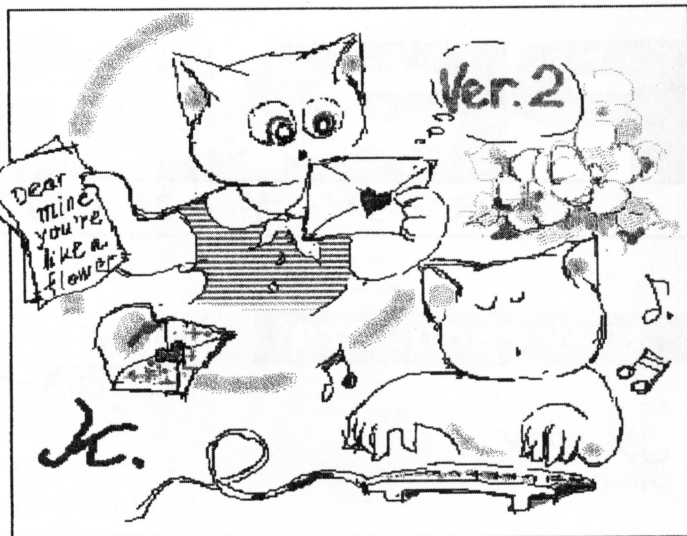
改造するための時間も技術もないまま，PTA活動はだいたいそのままつぎの年度に引き継がれるのが運命だが，私たちも同じように新しい委員の人たちに，しごとを引き継ぐときがきた。

しごとの内容を整理していたら，いやでも，自分がむやみにこしらえた印刷物をつぎからつぎへと目にすることになった。委員会での話し合いのための議事案もあるし，経過報告も連絡もある。みんな会合の直前に大いそぎでエディタでこしらえたもので，これを委員の数だけコピーして配っていた。

ほとんどが勤めを持っているお母さんたちなので，委員会ごとに全員顔をあわせることはむずかしい。そんなとき，欠席した人たちにも同一の内容を伝えることができる印刷物は，なかなか便利だった。これらは同時に議事録にもなったし，しごとの進展を示すよりどころともなった。当時はそれなりに役割を果たしていたと言える。

でもここで，ある時間をすぎて，ゆとりの目でこれらを見返したとき，ワープロなどでますますさかんになっているミニコミュニケーションとしての印刷物について，いろいろと考えさせられることも多い。

委員会のためにできてしまったプリント類を見ると，無作法もたくさん目につく。会議のテーマをかかげて，進行の目安にするはずの書類に，こまかな段取りや予想される決定事項まで書いてあって，審議の手間をはぶいてあったりする。時間にゆとりのない委員会であり，そのほうがおたがいに好都合な点も多いとはいえ，失礼なやりかただ。

「早く切り上げてもらえるので，助かるわぁ」と，お母さんたちはいっこうに異議を申したてる気配もないけれど，これでは会議とはいえない。そんな押しつけのやりかたを，なんとなく受け入れさせるのも印刷物のずるさかもしれない。

活字というものは，あらためて一方的な力を持っているものだと思う。それと矛盾するかのように，人に読ませる力がとぼしい面も持っているのも活字のようだ。たいせつな申し合わせ事項が書かれていても，読まれずにすごされたり，わざわざ電話で質問をしてくる人もいたりする。無個性の強みと弱みを持っているのが活字だ。

新聞，書物，雑誌，単行本，すべては活字からなる印刷物だが，読む者の数が大きくなるほど，個人的に受けるものは小さくなっていく。逆に，小さな範囲の中での印刷物ほど，自分への重さが増してくる。いちばん大きな重さを持つのが，特定の人にあてられた手紙ということになる。

いま，1対1でやりとりされる書簡をワープロ，パソコンでつくろうというときには，それなりの注意深さが必要になるはずだとあらためて思う。

英文タイプで手紙を打っていた歴史を持つ国の人たちは，活字に特別な思いもないかもしれないし，同時に控えができるやりかたは，OA機器にもそのままつながる方法で合理的だ。日本人は手書きですごしてきたから，基本的には手紙の控えというものはなかった。でも，手もとに痕跡が残らないところに手紙の使節としての夢があったようにも感じる。飛び去った飛行機に似た思いだ。手もとにファイルとして残されて，再編集する手紙は，往生ぎわが悪い。

私信のきわめつけ，ラブレターの自作がファイル管理されていて，苦い思いとともにたびたび読み返し，ついに編集を始めるなんて，おかしくて悲しい。

# PENGUIN INFORMATION CORNER

## ペ・ン・ギ・ン・情・報・コ・ー・ナ・ー

## NEW PRODUCTS

### カラーイメージスキャナ
### JX-220X
### シャープ

JX-220X

シャープはX68000のカラー入力機器として，カラーイメージスキャナの新製品「JX-220X」を発売した。

この「JX-220X」はコンパクトな薄型デトップサイズながら，キメ細かなズーム機能や色ずれの少ない1走査読み取りなどの基本機能を装備。さらに，高画質フルカラー出力，誤差拡散法，独自のディザ法による中間調再現や，各種ディスプレイ，プリンタに対応した濃度補佐機能などの高画質処理を備えている。

インタフェイスは汎用性の高いRS-232C，および高速伝達のできるX68000用パラレルインタフェイスを標準装備。また，付属のX68000専用ユーティリティソフトで容易に読み取り，市販ソフトへのデータファイル変換が行える。

さらに，専用ケーブルでプリンタに接続することで，読み取った画像を直接（コンピュータを介することなく）プリントアウトできる。

価格は168,000円（税別）。

〈問い合わせ先〉

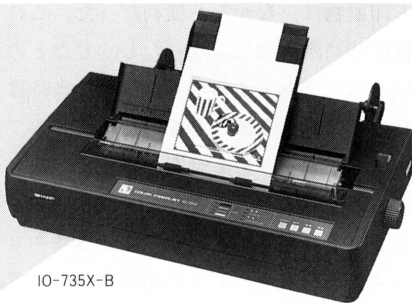
シャープ㈱　☎03(3260)1161,06(621)1221

### カラーイメージジェット
### IO-735X-B
### シャープ

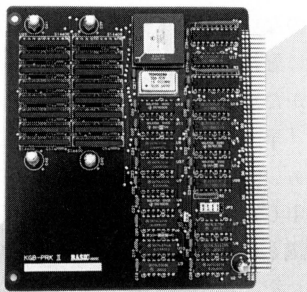
シャープは従来のカラーイメージジェット「IO-735X」に色調が追加されたモデル「IO-735X-B」を発売した。イエロー，マゼンタ，シアン，ブラックの4色各12ノズルを集積化した積層マルチノズルにより高速印字が可能。各色同時にプリントする方式のため，モノクロ/カラーが混在する文書や図形も印字速度が低下することなく，専用紙はもちろん，普通紙，OHPフィルムにも印字できる。制御コマンド体系は，G,P,Nの3モードを搭載しており，幅広い機種で使用可能。

価格は248,000円（税別）。

〈問い合わせ先〉

シャープ㈱　☎03(3260)1161,06(621)1221

IO-735X-B

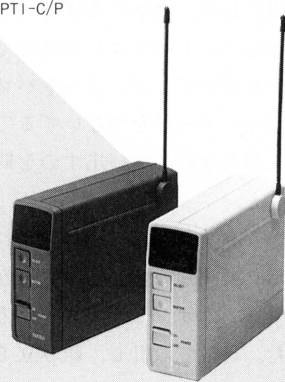
### 最大メモリ8Mバイト
### KGB-X68PRKII
### 計測技研

計測技研は，X68000用メモリボード「KGB-X68PRKII」を発売した。このボード上には最大8Mバイトまでの増設メモリを収めることができ，さらに数値演算プロセッサを搭載したモデルもある（なしのモデルでも搭載可能）。値段は2MBの55,000円から，8MB，数値演算プロセッサつきの190,000円（ともに税別）まで。メモリ容量2/4/6/8MBの4種類があり，それぞれに数値演算プロセッサ有無のモデルがある。

KGB-X68PRKII

〈問い合わせ先〉

㈱計測技研　☎0286(22)9811

### ワイヤレスプリンタターミナル
### 「PRINTER MATE」WPT1-C/P
### 積水化学工業

WPT1-C/P

積水化学工業が発売した，ワイヤレスプリンタターミナル「PRINTER MATE」WPT1-C/Pは，複数台のパソコンと複数台のプリンタを配線なしでつなぐネットワーク機器である。

無線という伝送媒体にはノイズの心配がつきまとうが，この製品では同社がFA，物流分野で築き上げたデータ伝送の信頼性を生かしており，ビット誤り率で$10^{-7}$以下という有線ネットワークと同等のデータ伝送信頼性を誤り制御ソフトウェアで達成している。

パソコン100台でプリンタ1台を共有可能で，プリンタが複数台ある場合は最大5台まで自動切り替えが可能。空いているプリンタを素早く選択してくれる。バッファもパソコン側256Kバイト，プリンタ側256Kバイト用意されている。

また，「WPT1-PX」はパソコンとの有線接続可能。プリンタ近くのパソコンとはこれまでどおりケーブルで，遠くのパソコンはワイヤレスで結ばれる。

価格はプリンタ側に接続する「WPT1-C」が70,000円，パソコン側に接続する「WPT1-P」が74,000円，「WPT1-PX」が79,000円（すべて税別）。

〈問い合わせ先〉

積水化学工業㈱　☎06(365)4504

## 電子手帳データネットワークシステム
# 電子手帳用通信カード&モデム
### シャープ

電子手帳用通信カード＆モデム

シャープは，電子手帳を利用して既存のパソコンネットを通じ，各種サービスを受けられる電子手帳ネットワークシステムを開発した。

本システムはハイパー電子システム手帳を使用するもので，通信用ソフトを内蔵した「通信カード」と「電子手帳用モデム」によって構成されている。

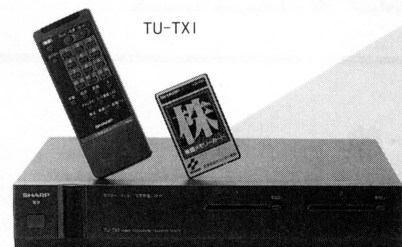
操作は，サンデーネット内にASCII-NET，EYE-NET，博多っこBBSへアクセスするための手順情報が登録されているので，簡単に行うことができる。ネットワークから取り込んだ情報の必要な部分のみを取り出して，電子システム手帳のメモに転送することなどももちろん可能。

「電子手帳用モデム」は電池による駆動，そして，携帯性にすぐれたコンパクト設計（タバコサイズ）になっている。通信速度は1200bps，ATコマンド対応，エラー自動訂正，MNPクラス５も搭載している。

発売は本年秋の予定で価格は未定。

〈問い合わせ先〉

シャープ㈱　☎03(3260)1161，06(621)1221

## 文字放送活用の情報システム
# TU-TX1&PA-9C81
### シャープなど

TU-TX1

コナミ，住商電子デバイス，日経テレプレス，シャープは共同で，テレビ文字放送を活用できる情報システムを開発し，７月10日に発売する。

「TU-TX1」は通常の文字放送を受信できるほか，ICカードスロットが備えてあり，付属の株価メモリカードを使えば，日経テレプレス（大阪地区は日経テレプレス大阪）の文字放送の株価番組を自動的に受信，蓄積する。蓄積したデータはチャート表示など多様な利用が可能。

また，それに加え，同時発売の文字放送日経テレプレスカード「PA-9C81」を利用することで，同文字放送局の他の番組（13番組）も受信，蓄積できる。このカードをハイパー電子システム手帳に装着することで，番組の内容を表示，活用できる。

これらの情報システムを利用することで，文字放送の情報を十分活用でき，特に電子システム手帳と組み合わせて使うことで携帯性に優れたホームインテリジェントターミナルの実現が可能になる。

価格は「TU-TX1」が90,000円（税別），「PA-9C81」が16,000円（税別）。

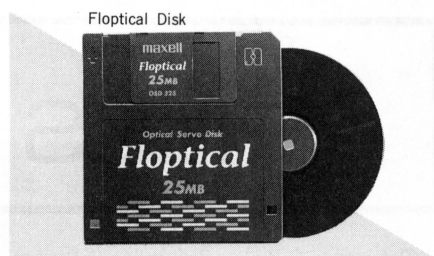
〈問い合わせ先〉

シャープ㈱　☎03(3260)1161，06(621)1221

## 光と磁気が融合した新しいメディア
# Floptical Disk
### 日立マクセル

Floptical Disk

日立マクセルは，光サーボ技術を取り入れて記憶容量25Mバイト（アンフォーマット時）を達成した3.5インチの新しいメディア「Floptical Disk」を今秋よりサンプル出荷する。

「Floptical Disk」は米インサイト社より要請を受けて開発したもので，同社の「Floptical Drive」（１台あたり50,000円程度）に適合する記録メディアということになる。このメディアは光サーボ・トラッキング技術に対応することにより，トラック密度，およびトラック本数を現行2HDに比べ，約10倍に高めている。日立マクセルとインサイト社は，このシステムを世界に広く普及させるとともに，将来に向けて大容量化（50Mバイト，100Mバイト）を目指すことで合意して，正式に技術提携を結んでいる。

サンプル出荷は９月からで，１枚あたり5,000円となる。

〈問い合わせ先〉

日立マクセル㈱　☎03(3241)9736

## 水中写真が手軽に
# 写ルンです　防水
### 富士写真フイルム

写ルンです　防水

富士写真フイルムは，水深３mまでの水中写真が気軽に楽しめるフジカラー「写ルンです　防水」を発売した。

フジカラー「写ルンです」は1986年に発売以来シリーズ化を充実しており，“手軽に，気軽に，簡便に”と好評を博している。今回発売される「写ルンです　防水」は水中や水辺での撮影が気軽にでき，水中での撮影操作をより簡便にするために，フィルムの巻き上げ部品なども大きめの設計がなされている。

これによって，フジカラー「写ルンです」シリーズは６種類９タイプとなった。

価格は2,000円（税別）。

〈問い合わせ先〉

富士写真フイルム㈱　☎03(3406)2111

# FILES Oh!X

このインデックスは，タイトル，注記——筆者名，誌名，月号，ページで構成されています。微妙なお天気のせいで，やる気をなくしがちなこの時期ですが，乗り切ればもう夏。頑張りましょうね。

## 一般

▶特別付録　ログインおまけディスク通信
MS-DOSフォーマットのディスクを付録。内容のデータは，X68000＆PC-9801シリーズ用シムシティーマップデータなど。——編集部，LOGIN，9・10号，付録

▶いま，シグオペ・シスオペになるには
パソコン通信のシグオペ・シスオペとはどんな仕事なのか，その現状はどうなっているのかをリポートする。シスオペの人の談話や大手ネットの申請資格一覧など。——編集部，パソコン通信，6月号，109-130pp.

▶東京新宿・新都庁舎
東京の新たなシンボル，新都庁舎。この建築設計にどんなコンピュータ技術が生かされているかを探る。——野沢潤一郎，マイコン，6月号，220-223pp.

▶MYCOM WATCHING
「渋滞を抜け出せるか！　走り始めた横浜市バス」と称してバスの路線状況を把握するのにコンピュータを取り入れた横浜市の試みを紹介する。——菊地秀一，マイコン，6月号，224-226pp.

▶入門ハード工作室
先月製作したロジック回路の拡張セットを作る。EXOR,EXNORをセットの中に組み入れ，さらに7セグメントLEDとそのドライバ，アップダウンカウンタなどを組み込んでさらに実験できる範囲を広げる。——石川至知，マイコン，6月号，315-320pp.

▶プロ野球DATA WATCHING
プロ野球界の野球記録データベースに対する取り組みを紹介する。日刊スポーツのN式データベース，ダイヤルQ$^2$を利用してメガドライブでアクセスできるプロ野球VAN,球団のデータ処理の方法などに迫る。——近藤攻司，ASCII，6月号，269-276pp.

▶C-TRACE CGコンペティション'91
キャスト主催のC-TRACE CGコンペティション'91の受賞作を紹介し，今回の傾向などを分析する。——編集部，ASCII，6月号，349p.

▶FREE SOFTWARE INDEX
ここ数カ月の間に主要通信ネットにアップロードされたPDSを紹介するコーナー。X68000用2HDフォーマット用デバイスドライバ，ディレクトリリスティングツール，MSXグラフィックローダなど多数のソフトが掲載。——編集部，ASCII，6月号，366-371pp.

## MZシリーズ

MZ-1500(BASIC MZ-5Z001)

▶きらきらHIWAY
でかキャラばりばりのカーアクションゲーム。——小枝直隆，マイコンBASIC Magazine，6月号，123-125pp.

MZ-2500(BASIC-M25)

▶HIDE'N DON
はいどんどん。相手のプレイヤーにブロックをぶつける，飛ばすなどしてダメージを与える。新しいアイデアいっぱいの2人用対戦アクション・ゲーム。——佐藤拓也，マイコンBASIC Magazine，6月号，126-129pp.

## X1/turbo/Z

X1シリーズ

▶ミケネコ印の宅配便
ベルトコンベアの上を流れてくる荷物を蹴り落とし，トラックに乗せる。——X1レタス，マイコンBASIC Magazine，6月号，153-154pp.

▶FOUR TRIS
上下左右からくるブロックをうまく組み合わせて正方形をつくる。パズルゲーム。——浅井新貴，マイコンBASIC Magazine，6月号，155-156pp.

▶ネオ投稿プログラムコーナー
「富士たかなすび道」というパズルゲームが掲載されている。同じ色のなすびが並ぶと消えてしまうという性質を利用して，画面上のすべてのなすびを消すのだ。——伊巻正，マイコン，6月号，346-357pp.

X1＋FM音源ボード（要NEW FM音源ドライバ）

▶アーケード版　パロディウスだ！　～ピエロの涙も三度まで～
ゲームミュージックプログラム。——桐畑厚宏，マイコンBASIC Magazine，6月号，186-187pp.

X1 turboシリーズ

▶カモーン
時間内に相棒のカモーンと積み荷，そして自分をロケットに乗せるゲーム。——松原拓也，マイコンBASIC Magazine，6月号，157-158pp.

## X68000

▶最新ゲーム徹底解剖!!
パロディウスだ！ の攻略ほかシグナトリー，ノスタルジア，サブナック，ファランクス，マーブルマッドネスなどを紹介。——編集部，LOGIN，9・10号，162-189pp.

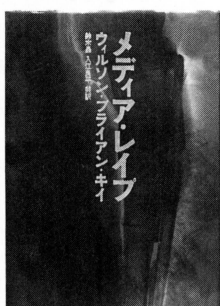
▶Software Review
メルヘンメイズ，続ダンジョン・マスター カオスの逆襲を紹介している。——X68000新聞社・白鳥ジュン，LOGIN，9・10号，190-193pp.

▶X68000新聞
XVI発売と同時にバージョンアップした付属ソフト「SX-WINDOW Ver.1.01」と「日本語ワードプロセッサ　Ver.1.1」を紹介。ゲームはA列車で行こうIII，プリンス・オブ・ペルシャ，ライヒスリッター，装甲騎兵ボトムズ，ネメシス'90。PDS紹介はMIDIソフト「STed Ver.2.0」——編集部，LOGIN，9・10号，244-251pp.

**参考文献**

I/O　工学社
ASCII　アスキー
コンプティーク　角川書店
THE COMPUTER　ソフトバンク
C MAGAZINE　ソフトバンク
テクノポリス　徳間書店
パソコン通信　エーアイ出版
POPCOM　小学館
マイコン　電波新聞社
マイコンBASIC Magazine　電波新聞社
LOGIN　アスキー

## 新刊書案内

メディア・レイプ　ウィルソン・ブライアン・キイ

かつて，本欄でも紹介したことがあり，有田氏も連載の中で取り上げていた「メディア・セックス」。その続編にあたるのが，この「メディア・レイプ」だ。中身の推測が不可能なタイトルだが，ようするにメディアを利用して人々の心を(当人たちに気づかれないように)コントロールするものの存在や，そういった術の存在を読者に警告するための書だと思えばいい。たとえば，シャンプーを売ろうと思ったら，自分のところのシャンプーの質の良さを宣伝するよりも，髪は常に清潔でなければならない，という信念を消費者に持たせるほうを採用する連中のことだ。

前作が徹底的に具体例を検証し続けたのに対し，本書はもっと広く，人間の教化されやすさ，脆弱さ，意識の裏側についてなどを，理論をまじえながら述べている。そういった点で，本書のほうが難しいところはあるが，面白い。

もし，あなたが自分の意志というものを信じているのなら，本書を読むべきである。そうでなくても，今の時代に生きていくなら，目を通しておくべきだろう。お勧めである。　(K)

**メディア・レイプ　ウィルソン・ブライアン・キイ著　鈴木晶/入江良平共訳　リブロポート刊**
**☎03(3983)6191　A5版　358ページ　2,575円**

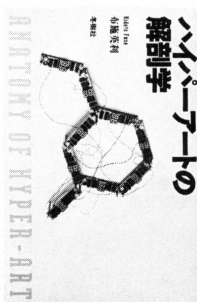
▶サイレント・フォース
スピードに自信のシューティングゲーム。――酒井優，マイコンBASIC Magazine，６月号，159-161pp.
▶SNAKE ACTION
サイドビューの，ふわふわ慣性つき面クリア型障害物よけタイム・トライアル・ワンキーゲーム。――福田圭介，マイコンBASIC Magazine，６月号，162-163pp.
▶TANK KILLER
マウスで照準を合わせ敵を撃て！　目指せ命中率100％のタンクアクション。――土屋達弘，マイコンBASIC Magazine，６月号，164-165pp.
▶疑似スプライト＆パレットアニメ処理
パソコン＆BASIC初心者向け講座。リストを載せて解説している。――おかちゃんぺ，マイコンBASIC Magazine，６月号，171-172pp.
▶アーケード版　パロディウスだ！　～ピエロの涙も三度まで～
ゲームミュージックプログラム。要NAGDRV＋CM-64。――野口英二，マイコンBASIC Magazine，６月号，190-194pp.
▶短期集中攻略　続ダンジョン・マスター
最終回・DAINの道の攻略。――石井弘一＆解せないクン，マイコンBASIC Magazine，６月号，269-273pp.
▶Ｘ68000芸術祭インフォメーション
ユーザーオリジナル作品を持ち寄って楽しんじゃおうというシャープ主催のＸ68000イベント「Ｘ68000芸術祭」の情報を掲載。――山下章，マイコンBASIC Magazine，６月号，279p.
▶GAMING WORLD
AⅢ（Ａ列車で行こうⅢ）マップコンストラクション，ヴェインドリーム，パロディウスだ！，スコルピウス，ループスなど。新作ゲーム先取りは，ファランクス，ザ・マジック・キャンドル，シューティング68Ｋなど。――編集部，テクノポリス，６月号，8-34pp.
▶software Hot Press
ファランクスやスコルピウスを紹介。――編集部，POPCOM，６月号，14-15，23pp.
▶ゲームの達人
AⅢマップコンストラクション，メルヘンメイズ，パロディウスだ！を紹介。――編集部，POPCOM，６月号，80-91pp.
▶Hard Laboratory
Ｘ68000XVIを徹底解剖。右タワーパネルを開けての中身の紹介や，高速化された付属プログラムのチェック。――編集部，POPCOM，６月号，100-101pp.
▶SOFT EXPRESS
パロディウスだ！，ノスタルジア，シグナトリーを紹介。――編集部，コンプティーク，６月号，70-79pp.
▶ONLINE SOFTWARE REVIEW
フリーウェアとしてはもっともポピュラーなミュージックドライバ「MDX」を紹介する。演奏しながらのテンポ調整や早送り・巻き戻しを備えた高性能なドライバ。鍵盤で演奏状況を表示するMDXS，詳しい曲情報を調べられるMDXSPなども同時に紹介。――小曽根雷太，パソコン通信，６月号，86-87pp.
▶X68000マシン語入門
対話型ソフトを作るシリーズの第３回。前々回のリストについての解説と，前回のリストの説明を行う。――高橋雄一，マイコン，６月号，277-284pp.
▶HOBBY EXPRESS
「パロディウスだ！」のゲームレビューを掲載。――あゆかわさつみ，マイコン，６月号，330-331pp.
▶なんでもＱ＆Ａ
X68000の新製品のソフトウェア面でのバージョンアップのポイントについてもインフォメーション，SWITCHコマンドに関する質問事項２つを取り扱う。――シャープ株式会社液晶映像システム事業部第２商品企画部，マイコン，６月号，376-377pp.
▶MYCOM TOPICS
X68000XVIのオプショナルパーツ発売のニュース。数値演算プロセッサ，2Mバイト増設RAMボードなどの仕様をアナウンスしている。――編集部，マイコン，６月号，391p.
▶この夏はメモリとハードディスクで決める!!
ASCII６月号特集。RAMボードとハードディスクの基礎知識と最新事情を紹介。X68000のRAM増設をめぐる環境も説明。――編集部，ASCII，６月号，213-236pp.
▶NEW MODEL IMPRESSION
５年目に突入したX68000の今年のニューモデル，X68000XVI/XVI-HDのハード・ソフトの概要を紹介。――編集部，ASCII，６月号，256-257pp.
▶PRODUCTS SHOWCASE
レイトレーシングによって3DCGを作成する「C-TRACE＋」を取り上げる。メタボールや，透過率を考慮するαチャンネル機能などを加え，パフォーマンスをアップしている。――編集部，ASCII，６月号，266-267pp.
▶AVプログラミング講座
３回に分けてポリゴンサーフェイス表示を行うプログラムを解説する。今回の講座は基本的な３次元表示の考え方について。――宮本親一郎，ASCII，６月号，309-316pp.
▶AV STRASSE
グラフィックとテキストを同一画面上でWYSIWYG式に扱えるマルチワープロ，Multiword PRO-68Kを紹介。PDSはテキストビュアview.x，ディスクエディタdedit.x，バックグラウンドタスクのサポートシステムBGDRV.Xを取り上げる。――仲田津宏，ASCII，６月号，317-320pp.
▶ハードディスクIPL SCSI版
ハードディスクをセットしているとフロッピーからの立ち上げがうっとおしい場合もあるが，それを解決するのがこのIPLプログラム。フロッピーとハードディスクの優先順位の設定や，起動領域の選択などが行える。――市原昌文，I/O，６月号，169-179pp.
▶PC USER'S GUIDE
この４月にニューモデル，XVIをラインアップ，SX-WINDOWもバージョンアップされてパワーアップしたX68000シリーズ。その性格と操作感覚をレポートし，性能のテスト，ライバル比較やインタビューなどを加えてX68000の実体に迫る。――田中利昭，THE COMPUTER，６月号，114-123pp.
▶GCCで学ぶ68ゲームプログラミング
X68000をベースにゲームを作りながら豊富な機能を使いこなすＣプログラミングを学ぶ立体講座。使用するのはGCCだ。第１回はスクロールに関するテクニック。――吉野智興，C MAGAZINE，６月号，96-100pp.
▶LHAX X68k
高圧縮書庫管理プログラムLHAのHuman68kバージョン。移植は岡田紀雄氏。――C MAGAZINE，６月号，付属ディスク。
▶One Point Edition
異なるOS間の移植には数々の問題がつきまとう。村田敏幸氏がMS-DOSからHuman68kへのＣプログラムの移植作業に関する注意点・問題点を解説する。――村田敏幸，C MAGAZINE，６月号，81-88pp.

## ポケコン

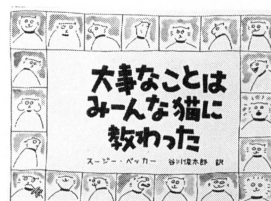
**PC-E500**

▶誌上公開質問状
PC-E500で作ったプログラムは，本体に記憶しておくことができるか？　また，マニュアルに載っていない関数はどんなものがあるか？　などの質問に答えている。――ドラゴン，マイコンBASIC Magazine，６月号，92p.
▶SUPER WAHAHA BOY
敵とバクダンを避けながら，ゴールを目指す。ギャンブル性も含んだゲーム。――高本栄一郎，マイコンBASIC Magazine，６月号，167-168pp.
▶SEVEN UP
テトリスタイプのパズルゲーム。全12面。――小野勝昭，マイコンBASIC Magazine，６月号，169-170pp.

**PC-1600K**

▶PC-1600K実践プログラミング
ポケコンを実務で使ってみようという人を対象に，簡単なプログラム例を交えて活用法を解説する講座。今回はプログラミングの基礎と，ポケコンの特長であるプログラムリザーブの活用法について説明する。――塚田洋一，マイコン，６月号，268-272pp.

**ハイパーアートの解剖学**

美術評論家である布施英利氏が雑誌に掲載した原稿をまとめたもの。だから，書評あり，評論あり，対談ありの幕の内だ。基本的にアートについて語っているのだが，その視点が非常に気持ちいい。我々は何を見て「リアル」と感じるか。つまるところ，「脳」である。わけのわからない美術論ではなく，美術といっても，ハイパーアートから音楽から写真，映画，CG，ビデオと幅広いので，安心して読める。（K）

**布施英利著　冬樹社刊　☎03(3543)4731**
**Ｂ６版　283ページ　1,600円**

**大事なことはみーんな猫に教わった**

猫の生態を描いた絵本である。本書を読むとなぜ猫のわがままが許されるかわかる。ということは，なぜ，猫のような女のわがままは許せて，犬のような女のわがままは許せないかがわかる。わがまま放題してなおかつ許してもらいたいなら，本書を読んで学ぶべし，谷川俊太郎氏の序文が本質をとらえていて○。（K）

**スージー・ベッカー著　谷川俊太郎訳**
**飛鳥新社刊　☎03(3263)7770　B6版**
**94ページ　1,200円**

# QUESTION and ANSWER

**Q** 最近，X-BASICを使ってゲームを作ろうとしましたが，速度に不満を感じてアセンブラを使おうと思いました。しかし，以下のことがよくわからないためプログラム制作がストップしているのでご指導よろしくお願いします。

**1)　本体付属のスクリーンエディタで漢字を打つにはどうしたらいいのですか。**

**2)　アセンブラにおける疑似命令（dc, text, evenなど）の使い方。**

**3)　スプライトを表示するときに必要なX座標とY座標の値をレジスタからメモリに確保したい場合どうすればいいのですか。**

**4）ジョイスティックのデータを取り出してスプライトの表示座標を変えるにはどうすればいいのですか。兵庫県　佐保　俊明**

**A** さっそく回答していきます。1）については日本語FEPのASKがシステム立ち上げ時に組み込まれていて，辞書が指定ドライブに存在していればCTRL＋XF1で日本語入力モードになります。もう一度CTRL＋XF1を押せば抜けられます。

2)の疑似命令というのは68000のオブジェクトコードではなく，アセンブラに与える命令ということを覚えておいてください。で，とりあえず最低限必要と思われる疑似命令について説明していきます。

.text……この命令以降の領域がプログラムであることを宣言します。普通，ソースプログラムの先頭に置きます。

dc.データサイズ……初期値を持つ変数を定義します。データサイズは，b,w,l（バイト，ワード，ロングワード）のいずれかです。

ds.データサイズ……使い方は，

```
     ds.w        10
```

のようにします。これは初期値を持たない変数を連続的に確保するための命令です。データサイズはdc.命令と同じです。上記の例はワードサイズのデータを10個確保します。

.even……68000では奇数アドレスからワード，ロングワードのデータが始まることは許されていません。ところがバイト長のデータが定義された場合に，後続のワード，ロングワードのデータが奇数アドレスから始まってしまうときがあります。そのような事態になったときにバイト長のデータの最後に1バイト付加してアドレスを補正させる命令です。とりあえずワークエリアの始まりと終わりのところに置いておけば問題ないでしょう。

以上を理解しておけばだいたい間に合うと思います。

3)，この質問については，いま説明した疑似命令を使います。X,Y座標を格納するX,Yという変数を定義したいのであれば，

```
     .text
       ⋮
 (プログラム)
       ⋮
X   dc.l   00
Y   dc.l   00
```

とします。で，定義したX,Yから値を取り出してIOCSコールのSP_REGSTを使いスプライトの表示を行いたい場合には，

```
     move.l   #$80000000,d1
     move.l   X,d2     ＊X座標の取り出し
     move.l   Y,d3     ＊Y座標の取り出し
     moveq.l  #0,d4
     move.l   #1,d5
     IOCS     _SP_REGST
```

とすればOKです。

リスト1

```
=================  sample.s  ==================
      1: *
      2: *スプライト移動ルーチン，サンプル
      3: *
      4:
      5:          .include       doscall.mac
      6:          .include       iocscall.mac
      7:
      8:          .text
      9:
     10: main:
     11:          move.w  #2,d1                *256*256 高解像度モード
     12:          IOCS    _CRTMOD
     13:          IOCS    _SP_ON
     14:
     15:          move.l  #16,x                *X座標初期化
     16:          move.l  #16,y                *Y座標初期化
     17: mloop:
     18:          bsr     wait
     19:          bsr     sp_put
     20:          bsr     joy_main
     21:
     22:          moveq.l #0,d1                *ESCキーで終了
     23:          IOCS    _BITSNS
     24:          btst    #1,d0
     25:          beq     mloop
     26:          DOS     _EXIT
     27:
     28:
     29: *ウェイト
     30:
     31: wait:
     32:          move.l  #10000,d0
     33: wait2:
     34:          subq.l  #1,d0
     35:          bne     wait2
     36:          rts
     37:
     38: *
     39: *スプライトの表示
     40: *
     41:
     42: sp_put:
     43:          move.l  #$00000000,d1        *スプライト番号
     44:          move.l  x,d2                 *X座標の取り出し
     45:          move.l  y,d3                 *Y座標の取り出し
     46:          move.l  #$00000100,d4        *パターンコード
     47:          moveq.l #3,d5                *プライオリティ
     48:          IOCS    _SP_REGST            *スプライトレジスタの設定
     49:          rts
     50:
     51: *
     52: *ジョイスティックの入力
     53: *
     54:
     55: joy_main:
     56:          moveq.l #0,d1                *ジョイステック0
     57:          IOCS    _JOYGET
     58:
     59:          moveq.l #0,d3                *Y方向の移動量
     60:          moveq.l #0,d2                *X方向の移動量
     61:
     62:          btst    #0,d0                *上方向のチェック
     63:          bne     jd
     64:          moveq.l #-1,d2
     65: jd:
     66:          btst    #1,d0                *下方向のチェック
     67:          bne     jl
     68:          moveq.l #1,d2
     69: jl:
     70:          btst    #2,d0                *左方向のチェック
     71:          bne     jr
     72:          moveq.l #-1,d3
     73: jr:
     74:          btst    #3,d0                *右方向のチェック
     75:          bne     jj
     76:          moveq.l #1,d3
     77: jj:
     78:          add.l   d2,y                 *Y方向の座標更新
     79:          add.l   d3,x                 *X方向の座標更新
     80:          rts
     81:
     82: x:       dc.l    00
     83: y:       dc.l    00
     84:
     85:          .end
```

4)の質問はBASICでプログラムを組めるのなら2),3)のことを理解すれば問題はあまりないでしょう。一応サンプルプログラム（リスト1）を作ってみました。実行すると画面の右上にパターン番号0のキャラクタが表示され，ジョイステイック0でキャラクタを8方向に動かせます。動きが遅いのはウェイトを入れているためです。ESCキーを押すと終了します。では解説していきます。プログラムの流れとしては，

a）　座標，移動量の初期化

b）　X,Y座標を取り出してスプライトの表示

c）　ジョイスティックからデータを取り出してX,Y方向の移動量をレジスタにセットする

d）　セットされた移動量を座標に加える

となっていて，b)，c)，d)を繰り返しています。ということで問題のジョイスティックの入力ルーチンの説明です。ジョイスティックから値を入力させたいときにはIOCSコールのJOYGETを使います。

使い方はd1レジスタに読み込むジョイスティック番号をセットしてJOYGETを呼び出します。するとd0レジスタに値を返してくれます。詳しいことは仕様を見てくれればわかるとおり，返り値にはスティックが上下左右のどこに倒れているかの情報，どのトリガが押されてるかの情報がビット単位で格納されています。

リスト中では順番に上下左右のビットを調べていきそれぞれX,Y方向の移動量をセットし，サブルーチンの最後でX,Y座標に移動量を加えて座標を更新しています。ポイントは上下左右，4方向の入力しかないのにプログラムではちゃんと8方向に動くという点です。よくわからないときには1行ずつ確かめながら見ていってください。

ということで質問に答えてきました。しかしこれらの質問は，ほとんどがマニュアルまたはX68000関連書籍に書いてあるものです。わざわざ質問箱に送られてきたということはそれらの本をお持ちでないと思いますので紹介しておきます。

『X68000ベストプログラミング入門』（技術評論社刊）

・X68000についてハードウェアの解説に始まりBASIC，C言語，アセンブラ，OS-9についてひと通りの解説がされています。

『X68000マシン語プログラミング』（ソフトバンク刊）

・本誌連載の記事を単行本化したものです。アセンブラによるプログラミング技法を中心に扱っています。

『68000プログラマーズハンドブック』（技術評論社刊）

・MPU68000の解説書です。泥沼のプログラム高速化に絶対必要になる，各命令のクロック数が載っています。

以上の3冊が特に有名です。全部揃える必要はありませんから書店で立ち読みをして自分に必要なものを購入してください。

（浜崎　正哉）

**Q　X1turboでturbo用“SWORD”を使用しています。文字色と背景色を変えたいと思い調べたところ，文字色についてはBIOSのワークF8D0$_H$にカラーコードを書き込めばよいことがわかりましたが，背景色はどのようにしたらよいかわかりませんでした。プログラムによって(コントロールキーを使わずに)背景色を変えるにはどうしたらよいでしょうか。**

**埼玉県　新木　健**

A　背景色を変更するためにはパレットコード0を変更します。S-OSから背景色を変更するのであれば，アセンブラでパレット変更プログラムを作れるようになりますね。リスト2を紹介しましょう。このプログラムはパレットを標準状態に設定するものです。

リスト2

```
=================  situ.s  =================
     1:
     2:          ORG      $8000
     3:
     4: PALET:
     5:          LD       BC,$1200
     6:          LD       A,$F0
     7:          OUT      (C),A     ; 緑
     8:          DEC      B
     9:          LD       A,$CC     ; $CDにすると背景色が赤になる
    10:          OUT      (C),A     ; 赤
    11:          DEC      B
    12:          LD       A,$AA
    13:          OUT      (C),A     ; 青
    14:          RET
```

図1

| カラーコード | 緑 12**$_H$ | 赤 11**$_H$ | 青 10**$_H$ | ビット |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 |
| 2 | 0 | 1 | 0 | 2 |
| 3 | 0 | 1 | 1 | 3 |
| 4 | 1 | 0 | 0 | 4 |
| 5 | 1 | 0 | 1 | 5 |
| 6 | 1 | 1 | 0 | 6 |
| 7 | 1 | 1 | 1 | 7 |
|  | F0$_H$ | CC$_H$ | AA$_H$ |  |

X1でパレットを変更するには，I/Oアドレスの10**$_H$,11**$_H$,12**$_H$にデータを設定します。これらは順に，青，赤，緑のデータセレクタになっています。起動直後のI/Oアドレスの状態を図1に示します。各データセレクタのビットが1で画面出力，0で画面出力を行わないようになっています。カラーコード3（紫）の欄を右に見ていくと，緑が0で赤と青が1であるので，ちゃんと紫になっていますね。

これを見てもらえばわかるように，パレット0を変更するには各データセレクタのビット0を操作すればいいのです。背景色を赤にするならカラーコード0の欄を0,1,0にすればいいのですから，書き込むデータは，

10**$_H$　F0$_H$

11**$_H$　CD$_H$

12**$_H$　AA$_H$

となります。

（影山　裕昭）

**質問にお答えします**

日ごろ疑問に思っていること，どんなことでも結構です。どんどんお便りください。難問，奇問，編集室が総力を上げてお答えいたします。ただし，お寄せいただいているものの中には，マニュアルを読めばすぐに回答が得られるようなものも多々あります。最低限，マニュアルは熟読しておきましょう。質問はなるべく具体的に機種名，システム構成，必要なら図も入れてこと細かに書いてください。また，返信用切手同封の質問をよく受けますが，原則として，質問には本誌上でお答えすることになっていますのでご了承ください。なお，質問の内容について，直接問い合わせることもありますので，電話番号も明記してくださいね。

**宛先：〒108 東京都港区高輪2-19-13**
**NS高輪ビル**
**ソフトバンク株式会社出版部**
**「Oh! X質問箱」係**

STUDIO X ON AIR

## FROM READERS TO THE EDITOR

**またまた，うっとうしい梅雨がやってきました。コンピュータにとってもあまり嬉しくない季節。毎日こまごまと手入れをしてあげましょうね。くれぐれもディスケットでカビなどを培養しないように注意しましょう。**

◆X68000 PROIIを買って9カ月。まだBASICしか使えません。Cコンパイラが高くて買えないためもあるが。ところで，Oh!Xの内容は難しい。しかし，毎月楽しく読ませていただいています。少しでも経験値をつんで内容についていけるようにがんばります。　坂倉　寛如(21)京都府

常に地道な努力が実を結びます。初心を忘れずにがんばりましょう。

◆X68000SUPERが出たときはたいして驚きもしなかったけど，X68000XVIはおおっという感じでした。私は16MHzになったことよりもコプロや増設RAMが内蔵できることのほうがうらやましい。　佐藤　毅(22)岩手県

そうですね僕はMIDIボード，増設メモリとSCSIインタフェイスをどうやって2つのスロットに押し込めようか悩んでいます。

◆シャープさん，私は来年の春を期待していますよ。あっと驚くようなマシンを頼みます。X68000が発表されたときのような感動をもう一度……。　植田　進(19)千葉県

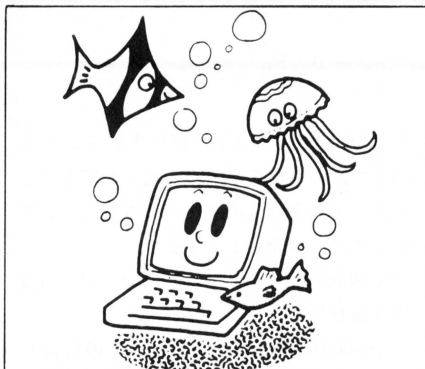
期待して待ちましょう。

◆ある本で読んだんですけど左脳を使いすぎるのはよくないそうです。いままで，頭を使えば使うほど刺激になっていいと思っていました。しかし，思考的なことを考えるのは左脳を偏重して使うので右脳とのバランスが悪くなってしまうからです。皆さん，右脳もちゃんと使いましょう。　岡部　誠(26)福井県

脳のバランスが崩れるとまっすぐ歩けなくなりそうで大変だなあ。

◆秋葉原のあるパソコンショップではX68000XVIの店頭デモを「沙羅○蛇」にしていた。うん，ユーザーのいちばん知りたいことをよくわかっているよい店だ。　佐藤　匠(19)埼玉県

とてもユーザー思いの店ですね。

◆ページをめくるたびに青ざめてしまった5月号。さて，私も買ってしまった“パロディウスだ！”。アーケードで見た瞬間，「これはX68000で出る」とピンときた私は一度も触れることなく待ち続けていました。出来は文句なしの2重丸です。細かいところに違いはある（らしい）ものの移植は完璧だと思います。ペンギンのテーマがお気に入り♪　まだ7面で大苦戦してますが絶対にクリア……したいなあ。　岩瀬　貴代美(19)福岡県

9面のペンギンたちはとってもかわいい。一見の価値ありですから見るまでがんばりましょう。

◆大学というところは非常に面白い。自分で書いた本を教科書だといい学生に買わせる教授，朝は苦手だということで講義時間を遅らせる教授，自分で出した数学の問題を解けなかった教授など実にバリエーションに富んでいる。なかには団体交渉権においては第1人者だという教授もいたっけ。　森下　昌仁(18)岡山県

名物教授というのはどこにでもいるものです。

◆最近パソコンには毎日触っているがゲームをするわけでもなく，プログラミングをするわけでもなくただいじっているだけである。いまいち無気力な状態になっているのだ。5月病ではなく1年中こんな感じである。どうしたんだろうか。　藤井　弘信(19)岡山県

何でもいいからやりたいことを見つけるようにするといいでしょう。

◆可愛い新入生たちよ，一生面倒見てあげるからね……と思ったら，なんと我々の後輩たちは全員男なのでした。困っちゃうなあーもう。　岡村　克宣(19)北海道

こうなったらバラの園を目指して，かわいい後輩を集めるのもいいと思います。

◆私も高校3年生となり，選択の授業で生物を選んだら男子が自分ひとりであった。隣の物理は男だけ。で，物理の授業を選択している野郎はハーレムだの両手に花だのいっているが，私にいわせると食虫植物の中の虫，または蛇に睨まれたカエル君の心境であります。　市川　徳明(17)東京都

やっぱりうらやましいです。

◆毎度お世話になります。ついに会社員というものにクラスチェンジしてしまった。いまは研修中なのでレベル0ってところかな。持ち物は布団と着替えぐらいしかなくてパソコンやオーディオはレベル1までお預けである。　田中　雅俊(24)神奈川県

研修中に大ポカをやらかして元のクラスに戻らないようにね。

◆とうとう社会人になってしまった。今は研修で楽しい授業だけど5月からは実務，あ〜，やってけるかなあ。それにしてもパソコンを触れる日が少なくなった。まったく時間が取れない！　橋本　広治(22)神奈川県

暇は自分で作るもの！　と意気込んでみても現実はうまくいかないもんですよね。

◆専門学校に入学して1カ月がたちました。学校の授業のおかげだかわからないけどHuman68kのコマンドが使えるようになった。それとC言語を習っている。これは楽しい。しかし疲れるので家に帰ってきてもX68000を触る気にもゲームをする気にもなれない。はあ。　宮下　誠(19)長野県

せっかくのX68000を活用するため，学校で習ったことを役立てるようにがんばろう。

◆ちょっとした手違いでOh!Xを2冊買ってしまった。これでディスク2枚組だぜー。　笹本　昌訓(24)山梨県

今度は4冊買ってディスク4枚組にしよう。

◀小川　裕美　山口県
小川さん，いつもたくさんの投稿ありがとう。なかでもこのハガキが元気いっぱい動きがあっていちばんよかったですよ。

◀鈴川　美佳子　東京都
相変わらずワイルドな絵柄。ちなみに手前のアジジおねーさまがアレックスおじさまをいじめているように見えるのですが。

◆やっぱりMAGICでしょう。以前，X１で必死に打ち込んだものだったけど，ついにX68000に移植されるとは感動しました。これからの発展の可能性を残しているし，サンプルゲームのSIONもよかった。誰かこれを使って「ジェルダ」や「ヒロトンウォーズ」でも作ってくれないかなあ。　池谷　尚紀(22)静岡県

他力本願はいけませんねえ。もっと積極的にアプローチしなくちゃ。

◆SIONにはバグがある。画面の奥に消えていく敵を画面から消えた直後に破壊すると，数万数千点も入ってしまう（デバイスドライバの有無や登録順によって違う)。これを利用したときの最高点は9792＋65536×4点！　利用しなかったときは33290点。　伊藤　浩克(19)香川県

さすが伊藤さん，目のつけどころがシャープだねえ。まいったまいった……。

◆「黄金週間PRO-68K」ありがとうございます。これで付録ディスクも３枚目になったわけですが，いつも内容の濃さに感激しております。これからもプログラミングすることの楽しさを伝えてくれる雑誌としてがんばってください。　坂田　肇(38)滋賀県

一部の人たちだけでなく，多くの人たちがプログラミングの楽しさを知ってくれるとうれしいですね。

◆高速グラフィックパッケージ「MAGIC」がすごい。観念してアセンブラを勉強しなくてはいけないな。それにしても付録ディスクのすばらしさに感激しました。どうもありがとうございます。　吉田　勝晶(18)群馬県

影山氏の努力により「MAGIC」もBASICから使えるようになりました。活用してね。

◆うーん，ディスクだとソースを打ち込まなくていいので楽だなあ。そのぶんじっくり解析できるし。SX-WINDOW関係のプログラムもソースつきなのでとても参考になります。話は変わるけど，私は最近１日48時間という不規則な毎日を送っています。おかげでたまに手が震えたりします(やばいかな)。でもこの不規則な生活を規則的に続けて人よりも８時間分多く活動してやるぞ。ハッハッハ（朝なのでちょっとハイ)。　橋本　忍(20)埼玉県

あまり無理をしすぎると体を壊してしまいますからほどほどにしておきましょう。

◆５月号のmicroOdysseyで付録ディスクのありかたについて考えさせられた。「もっともだ」と思う点と「そうかな？」と思う点があり，メディア出版側もまだ暗中摸索しているのだなあ，という感を受けた。　佐藤　博之(22)北海道

どういう位置づけで付録ディスクが存在しているか皆さんも考えてみませんか。

◆「黄金週間PRO-68K」ということだが，５月号が発売になる４月18日はまだゴールデンウイークではない。よって５月号のディスクの名前は「もーすぐ黄金週間だよ～ん。わくわくPRO-68K」が正しい。もっとも１週間早くなった締め切りに追われる編集部の方々にとっては「な～にが黄金週間でぇ！　けっ！　PRO-68K」なんだろうが。　柳井　敏彦(32)愛媛県

どんなにつらくとも読者の皆さんに喜んでいただければそれで満足なんです。

◀八重樫　恵　宮城県
季節はずれのイラストのような気もしますがどうやら宮城県はまだ冬のようです。瞳の処理に気をつけるともっとよくなりますよ。

◀荒田　靖明　神奈川県
こちらはさわやかな印象のイラスト。初投稿ということですがきれいにまとまっています。常連目指してがんばってください。

◆やった～ついに1Mバイト増設してやったぞ！　これでメモリを気にせずにサンプリングファイルを読み込めるし絵も描ける。しかし，その資金がYAWARAの仕上げで得た金というのはちょっと情けないかな。１枚200円（修正・トレース彩色)で150枚ははっきりいって地獄でした。ところで今のアニメ業界は人手不足らしくて，動画とか仕上げのほとんどが東南アジアや中国なんかでやっているそうですね。あの○ッド○ハウスも仕上げの80％はソウルでやっているらしい。今までなんとなく見ていたアニメでしたがそれを知ってからは，瞬きするのも恐れ多いという感じで見ています。特番やナイターでYAWARAが見れなくても「うんうん，よかったね」と思ってしまいますし，どんなに前回のフィルムを使い回しても，止めセルのカットが多くとも「そうか，きっと仕上げが間に合わなかったんだなあ」とおおらかに笑ってしまえるようになりました(あぶねーな)。編集部でもアニメの好きな人，ぜひ一度仕上げをやってみてください。　平　智征(22)神奈川県

だからといって妥協ばかりしていると結果はどんどん悪い方向へと進んでいってしまうので流されすぎないようにね。

◆ナディアが終わって悲しいよお。あれ？　Oh!Xってアニメ誌じゃなかったんですか。　宮越　良幸(19)神奈川県

似たようなところはあるかもしれませんが実は違うんです。

◆メカトロニクス制御はとてもありがたかった。マッピーキットが壊れてしまったとき，この記事どおりに誘導電流防止回路を組み込んで直したところ壊れなくなった。大会当日の朝までかかって徹夜でプログラムを組み上げたけど，デバッグが完全でなかったので心配していたが結果は意外なもので優勝してしまった。これもみなハードウェア工作のおかげだ。　熊谷　彰(18)宮城県

いやいや，それは君の実力です。それにしても優勝とはすごいですね。

◆「これから載せてほしい記事ってある？」と弟に聞いたら，

1)　音楽の才能がなくてもなんとなく曲らしいものが作れる講座

2)　絵がド下手でもZ'sSTAFFのさまざまな機能を使ってそれなりの絵が描けるようになる講座

といっていました。でもこれらは少しは音感とか絵心などがないとだめなものだと私は思ったりするのですが。　谷口　有香(21)北海道

両方とも自分で努力しているうちになんとかなるものですが，要は楽しくできればいいのだと思います。

◆ちょっと聞きたいのですが，

1)　満開の電子ちゃんのコミックはまだなんですか？

2)　なんで15,6歳の方々はあんなに絵がうまいのですか？

3)　やっぱりプレゼントも女性が優遇されるのですか？

4)　どうしてこんなにおなかが減るんですか？　別に答えてくれなくてもいいですよ（笑)。　横田　隆史(20)千葉県

行数の都合によりお答えできません。なんてね。

◆見ましたよ。５月16日放送のNHK，現代ジャーナル「文化情報」CGアートというものにOh!Xの表紙などでおなじみの須藤牧人氏が出演していました。CGが出た瞬間に「あっ，Oh!Xの表紙だ！」と大きな声で叫んでしまいました。ちなみに出ていたのは1990年の６月号と1991年の２月号と４月号のものでした。　山崎　勘太郎(18)愛知県

僕も見たかったな。

◆今日，自分のアパートに戻ってみると珍しいことにどこからかエアメールが届いていた。海外に知り合いがいる僕にとってこんなに嬉しいことはない。さっそく手紙を開けて読もうとしたが見慣れない文字が並んでいて，こりゃ訳すのは大変だなあと思っていたら，友達に「これ，お前のとこに来た手紙と違うじゃん」といわれた。よく見たら隣の外国人宛に来た手紙だった。あわてて隣のポストに入れたのはいうまでもな

い。　　　　　　　　稲松　清澄(22)神奈川県

もしも開封してしまったら，隣の家に来たお歳暮を間違って開けて食べてしまったカツオ君の心境を味わえたことでしょう。

◆メルヘンメイズのファンはあんなにいたのか？　某友人はメルヘンメイズを買った私に「とうとうそういうソフトを買ったか」といった。あたしゃ無実だ〜。大野　大輔(19)秋田県

本当にそういいきれますか？

◆他人事だと思っていたらとうとう今年は自分も受験生ではないか。そこで提案です。12月号あたりで「受験対策PRO-68K」という付録ディスクをつけてもらいたいのですがどんなもんでしょうか？　　　　斉藤　淳三(18)神奈川県

ついでに「夏季講習PRO-68K」，「直前模試PRO-68K」もつけてあげましょう(ウソ)。

◆現在，X68000初代に付属していたアセンブラでゲームを制作しています。以前にも1本ゲームを作ったのですが，ゲーム作りは「技より根性」だと感じます。ハイ。

徳丸　正巳(21)鳥取県

何かを作り上げるために最後にものをいうのは根性ですからね。

◆僕はこの間，なにげなくOh!Xの表紙の白いところを消しゴムでこすった。そしたらすごくきれいになったので僕は喜び，全部きれいにしてやろうと思い，端からきれいにこすっていった。しかし，絵のところをこすった瞬間，見事に色がはげてしまい僕はあ然とした。が，それも長くは続かなかった。Oh!Xをマウスパッドとして使っていたことを思い出したのだ。急いでマウスをひっくり返すと案の定マウスはOh!Xに染められていた。ちゃんとマウスパッドを買っておけばよかった。　　　松井　昭宏(14)三重県

ふっ，まだまだ若いのお。

◆家のCZ-652Cが可愛くて可愛くてしょうがない。こんな私を人は「メカフェチ」と呼ぶが，私はそれに対して「私はプラトニックだ」と答えることにしている（救いようがない……)。

薄井　広樹(21)北海道

いずれにしても愛があれば……。

◆ソニーのデータディスクマンを買いました。映画が好きなので「ぴあ」発行の「シネマクラブCDブック版」を買ったのです。しかしこれが大変なタコソフト。タイトル名の入力は濁音半濁音は受けつけない(「スターウォーズ」は「すたあうおおす」と入力)。また監督名では検索できても出演者では探せない。ジャンル別，原題名での検索もできない。これではせっかくのCD-ROMも泣いている。悪貨は良貨を駆逐する，の例にあるようにこんなソフトが今後出回ってきたら，もうソフトを買う気なんてなくなってしまう。やはりひとつのメーカーがソフトの質に関して検閲すべきではないのか。

大崎　芳美(28)愛知県

優秀なソフトウェアプレイヤーでも肝心のソフトが腐っていてはしょうがないですね。

◆最近パソコン通信でもやってみようかと思っています。初心者用の入門特集や活動状態を伝えるページを作ってもらえませんか。最近はMIDI関係が多いようですが若い人が多いからかな。　　　　外池　常彦(46)埼玉県

確かに若い人が多いですが子供と一緒に楽しくパソコンを触っているお父さんもちゃんといます。

◆今度，MIDIを買おうと思いそのためにバイトを始めたのですが，なんと1カ月ほどでバイト先が潰れてしまいました(悲しい)。ちきしょう買う日が遠のいてしまった。しかし，この次は買うぞ！　　　　中根　雄一(19)茨城県

早く次のバイト先が見つかるといいですね。

◆私はX68000を買ったときに友人の「X68000ならMIDIで音楽をやるのが一番！」というひと言でMIDIボードを買ったがいまだ音源がない。ああ，「CM-32L」があれば「パロディウスだ！」がグレードアップして遊べるのに……。

小沢　一成(17)東京都

さあ，君もがんばってバイトに励もう。

◆酒井さん，「ファイヤークリスタル」では「ホウセキ」はなんの役にも立ちませんしブラックオニキスの暗号も使いません。戦死はレベル10ならそれ以上強くなりません。僕はなんとか自力で解くことができましたがはたして最後まで解いた人って何人くらいいるのでしょうか？

黒武者　健一(21)神奈川県

返答が遅くなってしまいましたけど参考になりました？　酒井さん。

◆5月号の佐々木さんへ。ウチのX68000のキーボードはコンパウンド（研磨材）で磨かれてピカピカになっています。本当はキーボードの上についた傷を消そうとしたんですが……。

澤田　裕史(15)神奈川県

やっぱりキーボードだけだと違和感があるでしょうからディスプレイと本体も磨いちゃいましょう。

◆初めてアンケートを出します（変なイラストは出していますが)。ところで世間一般の皆様は「カオスの逆襲」はもうクリアしてしまったのでしょうか。私は最近，ゲームをプレイする時間がないのでパーティが路頭に迷っています。攻略本は読まない主義なので「ダンジョン・マスター」と同じくクリアに1年かかりそうです。ホエ。でも楽しめればいいですよね？

鈴川　美佳子(18)東京都

無事，エンディングが見れたら，また報告してください。

◆東京からの帰りついでにアルバイトへいくことにした。高速料金（1,600円）を節約するために一般道で帰ろうとしたら渋滞に巻き込まれて，アルバイトに2時間（−2,000円）遅れてしまった。さらにいつもは自転車なのにこの日は車で直行して路上駐車しておいたので，もしやと思って車を見るとしっかりステッカーが貼ってあった(違反点数2点，−15,000円)。1,600円を節約しようとした私の試みは15,400円の赤字を出してしまった。「パロディウスだ！」が遠ざかる。

野田　博(20)群馬県

そんな不幸はものともせず元気に生きてください。

◆うちの大学には生協がない。が，売店はある。そこには毎月Oh!Xが3冊しか入らないので18日は誰よりも早くOh!Xを確保しなければならない。早起きのかいあって今月は俺の勝ちだ！

榎本　和義(20)埼玉県

来月からはけんかをしなくてすむように店の人に頼んでOh!Xを20冊ぐらい入れてもらいましょう。

◆5月18日の午前12：00，私はボンバーマンに夢中になっていた。8-8（最終面）でなんとボスといろいろな色のボンバーマンの手下が出てきて，自機（白ボンバー）めがけて突撃し始めた。私は必死に迎撃したが敵はしばらくすると変身して炎を投げつけてきた。そして自分がセットした爆弾に引火して白ボンバーは死んでしまった。しかし私は再び敵ボンバーに挑んだ。うまく黒ボンバー以外の敵ボンバーを叩き潰して黒ボンバーとの対決になったが，何と黒ボンバーはバリアを張ったりワープしたりして白ボンバーを困らせた。無念にも白ボンバーは力尽きてしまった。8-8面は難しい。ボンバーマンのプロと自慢できるユーザーの皆さん，私に8-8面の攻略法を教えて。　　　大田　哲夫(19)神奈川県

編集室ではS.K.氏がボンバーマンのプロといわれています。彼はノーコンティニューで全面クリアした強者。今度会ったら攻略

◀清水　健太郎　静岡県
メカの描写がなかなかうまいですね。細かいところまでよく描き込んでいます。今度はパロディにも挑戦してみては？

◀岡村　直也　兵庫県
年一回連載，謎の姉弟4コママンガの岡村君の登場です。今度は月一回連載目指してがんばりましょう。ところでマクシヴィってなんですか？

法を聞いておきますね。

◆5月11日シャープのパソコンフォーラム'91に行ってきました。ビジネス中心のかたい内容を想像していたのですが行ってみると意外に親しみやすい内容でした。本当はもう少し情報収集になるような専門的な内容を期待していたので，ちょっともの足りない気もしました。しかし，パソコンの楽しさ，X68000ってこんなに面白いんだぞという感じが伝わってくるようでとてもよかったです。DŌGAの3Dアニメを実際に見れたのもよかった。これを見て自分もいつか3Dアニメをやりたくなりました。残念なのはSX-WINDOW対応グラフィックソフト「Easypaint」。ウィンドウは狭いし，いまいち使いにくい感じがしました。

ゲームコーナーはほとんどゲームセンター化していましたけど，こういう機会でないと試しプレイはなかなかできないわけだし貴重だと思います。しかもソフトハウスの人と話ができるチャンスなんて本当に貴重だと思いました。私はやっぱり凡人ですからゲームソフトの安売りが気になったり，X68000グッズコーナーで電飾POPが売れているのを見て妙に感心してしまいました。そうそう，新製品のビデオレセプターはいい！　ほしいけど高い！　の見本のようでした　　桜井　直樹(?)神奈川県

ちょっと会場が狭い気がしましたけどユーザーのオーラが伝わってくる楽しいイベントでしたね。

◆今年になって国語の授業を受けショックを受けた。去年は国語の授業といえば紙飛行機が飛びかい，窓際のものが先生の顔めがけて下敷きを使って光を反射させ，ウォークマンを聞き，弁当を食べ，挙句の果てには先生のスリッパが飛びかっていたのに……。今年はなんて静かなんだ。　　円福　貴光(17)愛知県

みんな来年の春には幸せな気分を味わいたいのでしょう。

◆私の視力は0.008以下である。そのおかげでハイテクを駆使した100グラム近い眼鏡をかけている。それでも0.3ぐらいしか視力がない。眼鏡を外すと目から3cm手前の指紋が見える程度である。みなさん視力はぜひ大切に（まだ17歳だというのに）……。やっぱりゲームが悪いのかなあ。　　山口　剛(17)東京都

いくら好きなこととはいえ，ほどほどにしておきましょうね。

◆いまだにX68000のソフトでPC-9801からのまるごと移植ってのがありますね。画面も640×400で作ってあって音楽も6重和音なのですぐわかってしまう。あとクロック速度だけ速いPC-9801のソフトをX68000にそのまま持ってくればどうなるか明白である。改悪ソフトはお断り。　　山田　慎也(21)神奈川県

せっかくの名作が台無しになるのは問題ですね。

▲見浦　崇　長野県
今度はX68000の元を取るまでガンガン使いこなしてやりましょう。何かいいものができたら投稿してね。

# ぼくらの掲示板

●掲載ご希望の方は，官製ハガキに項目(売る・買う・氏名・年齢・連絡方法……)を明記してお申し込みください。
●ソフトの売買，交換については，いっさい掲載できません。
●取り引きについては当編集部では責任を負いかねます。
●応募者多数の場合，掲載できない場合もあります。
●紹介を希望されるサークルは必ず会誌の見本を送ってください。

## 仲間

★パソコン通信をやっていて2週間に一度は連絡の取れる方，友達になってください。連絡待っています。〒438　静岡県磐田市中泉1282-56　鈴木　貴久(17)

## 売ります

★ローランド純正キーボードスタンドKS-8（定価9,000円）を送料込み3,500円で。D-10等，ほとんどのシンセに対応しています。連絡はハガキでお願いします。〒284　千葉県四街道市和比254林田　和也(19)

★GSM2400（2400bps.モデム）を1万円で。箱，付属品，説明書すべてあり。連絡はハガキで。〒207　東京都東大和市中央1-1140-8　秋本　乾太郎(18)

★ファクシミリ付きイメージスキャナプリンタMZ-1V01を6万～7万円＋送料で。新品同様，取扱説明書，箱付き。連絡はハガキで。〒989-12　宮城県柴田郡大河原町堤字南16　猪野　亨(22)

★X68000用イメージユニットCZ-6VT1-BKを35,000円以上で，プリンタCZ-8PG1を65,000円以上で売ります。高く買ってくれる人を優先します。完動美品，付属品すべてあり。詳細および連絡は往復ハガキにて。〒526-01　滋賀県東浅井郡びわ町落合726　大森　康雄(24)

★X1/X68000用熱転写カラー漢字プリンタCZ-8PC2を送料込み25,000円で。完動品，箱なし，説明書あり，黒・カラーリボン合わせて数本と感熱紙数百枚を付けます。連絡は往復ハガキで。〒120　東京都足立区千住元町34-4-603　島田貴宏(18)

★X68000用80MBハードディスクIT X680（アイテック製）を75,000～80,000円で。昨年9月に買ったものです。保証書，箱，説明書あり。連絡は往復ハガキで。〒430　静岡県浜松市竜禅寺町539　鈴木　幸太郎(20)

★X68000用カラーディスプレイCZ-606D-BKを送料込み37,000円で。マニュアル，保証書あり，箱なし。ほとんど使っていません。連絡は往復ハガキで。〒281　千葉県千葉市園生町1197-10　大貫　研二

★X68000用熱転写カラー漢字プリンタCZ-8PC4（グレー）を送料込み45,000円で。新品同様，箱，マニュアルあり。連絡は往復ハガキで。〒990　山形県山形市城南町1丁目16-63-534　斉藤　直史(16)

★X68000用熱転写カラー漢字プリンタCZ-8PC4を送料込み35,000円以上で。箱，付属品あり。高く買ってくれる人優先。連絡は往復ハガキにて。〒343　埼玉県越谷市蒲生西町1-3-52　青柳将司(18)

★インクジェットプリンタIO-735Xを8万円で。箱，マニュアル，付属品すべてあり。X68000用接続ケーブル付き。送料はそちら持ちで。連絡は往復ハガキでお願いします。〒680　鳥取県鳥取市胡山町東1-130-56カサブランカ103号　友国　直樹(20)

## 買います

★X68000用1MB増設RAM CZ-6BE1を送料込みの12,000円で。完動品で付属品ありにかぎります。連絡は往復ハガキで。〒093　北海道網走市潮見11-12-1-203　長縄　直樹(17)

★2400bps.モデム（完動品のみ）を送料込み1万円ぐらいで。できれば箱，マニュアルあり。キズあり可。価格と機種名を書いて往復ハガキで連絡してください。〒312　茨城県勝田市馬渡2655-2向野アパート2-31号　上山　潤一(17)

## バックナンバー

★Oh!MZ1987年3月号を送料込み1,200円で買います。切り抜き等不可。連絡はハガキで。〒632　奈良県天理市小島町89　龍見　将民

# 編集室から from E・D・I・T・O・R

## DRIVE ON

このコーナーでは，本誌年間モニタの方々のご意見を紹介しています。今月は5月号の内容に関するレポートです。

●数値演算プロセッサが宝石箱に入っているという話には大爆笑。さすが「目のつけどころがシャープ」ですね。たしかに「石」ですからねえ。そんなことはともかく，新しいFLOAT2.Xがほしい（のどから手が出てる）。また，最近レイトレーシングに凝っているので余計に16MHzがうらやましい。「コプロ積めば？」という声が聞こえてきそうだけれど，カメラもほしいし(写真にも凝っている)。うーん，16MHz……，68030がいいよお！

付録ディスクでは「SION」のBGMがよかったです。SX-WINDOWのアプリケーション，「SX風船」は「こりゃまた，やってくれましたわね」です。ライフゲームはたしかに解析すれば勉強になりそうですが，いちいちダイアログが出るたびに「ぴんぽ〜ん」とやってくれるのは，うっとうしくてしかたありません。

安井　百合江(16)　X68000 PRO　愛知県

●「MAGIC」はいいですね。最近はポリゴンばかりもてはやされていますが，ワイヤーフレームのほうが味があっていいと思うのです。ただ，使用可能言語がアセンブラということもあり，実際に使用するのはまだまだ先になるでしょうが，いつの日か実際にこの手で使ってみたいと思います。どうせなら「MAGIC」がカードゲームシリーズのように，毎月投稿作品が載るようになるといいのですが。

また，「言わせてくれなくちゃだワ」での「all that's Bug '90」は，12月号の「Oh!X INDEX '90」と同じ号にあったほうが，バックナンバーを購入したり，昔の記事を探すときなどに便利なのでは，と思います。

高橋　毅(19)　X68000 PRO,MSX2　埼玉県

●「言わせてくれなくちゃだワ」は昨年とほとんど変わらない構成でしたが，Oh!Xのおキマリですから，このままでいいのではと思います。京都の祇園祭りで毎年内容が変わるでしょうか。おそらく“伝統”にもとづいてことが運ぶでしょう。そうです。伝統は大事なのです。Oh!Xには伝統として“ドラゴンだ”の精神にのっとった，“難しい内容の記事でも平気で載せていくよ”ということのほかに，遊びゴコロの伝統としての“ちゃだワ”を守っていくのもよいのではないでしょうか。“伝統”は“マンネリ”につながるものではないはずです。

付録ディスクについては，私はX68000ユーザーではないのでわからないのですが，解析するのが楽しそうなプログラムがたくさん入っています。だから，いつかX68000を手にしたときのために，付録ディスクはすべてとってあります。

梅本　英之(21)　X1G，PC-8201，PC-1251　奈良県

●いやあ，「REAL」ですか。すごいのひと言につきますね。私なんかはダンプリストを入力するのは得意なんで，6時間ほどで入力してしまって，とりあえずキャラクタを8方向に走らせて遊んでみました。「SLANG」より遅いということですけど，実数を扱うのにかかる時間を考えればこのくらいは痛くないという次元でしたし，最高です(でも，それは「SLANG」の「SOROBAN」呼び出しが@ではじまっていたせいのような気もする)。

高村　信(20)　X1turbo,PC-8001mkII　東京都

●「REAL」について。ついに実数演算のできる「SLANG」（少し違うが）が大貫氏の手によって発表された。が，素直に喜んでよいものだろうか？　たしかに大貫氏はOh!Xの読者で，「SLANG」，「SOROBAN」の作者である。しかし，「不満があれば自分たちで直していく」のがS-OSではなかっただろうか？　そのためのソースリスト公開なのだ。できることならほかの人からの投稿であってほしかった。

さらに，企画に対する意見，要望も減っているとのこと。僕自身「SENTINEL」であるはずなので，あまり人のことはいえないのだが，もっとしっかりしなければと思います。最後に大貫さんご苦労さまでした。

奥村　光雄(16)　MZ-2000　埼玉県

●「大人のためのX68000」で，やはりプリンタユーザーのいちばんの問題になるであろうプリンタ用紙について書かれていたのがよかった。プリンタユーザーにとって金がかかるのが，インクリボン，カートリッジ，そして部屋のあちこちに散らばる無駄になったプリンタ用紙です。プリンタを使うときには，気をつけないとお金，そして紙の無駄遣いになるので，この荻窪さんの記事でいま一度考え直さなければならないだろうと思った。

段　宏太郎(20)　X68000 EXPERT　福岡県

●私が今回の特集で特に注目した記事が「速くなったFLOAT?.X」でした。いままで，FLOAT2+.Xを愛用していた私は，考え方を改めさせられたのです。シャープ系のマシンは，よりよい環境をメーカーに求めるのではなくて，ユーザーパワーで改善していくのが常でしたし，またユーザー自身もそう思っている節がありました。これはFLOAT2+.XやTurbo Consoleの存在を完全に肯定するものです。

しかし，Turbo Consoleを超える形でIOCS.Xが，FLOAT2+.Xの場合には新FLOAT2.Xが，どちらもメーカーの手で用意されてしまいました。なんだかんだいってもやっぱり最後にはお金も力もあるメーカーがすべてやってくれるようになるのでしょうか。

でも，よく考えたら，このFLOAT2.Xは，X68000 XVIを買った人だけが持っているものだったりするのです。シャープさん，早く従来機種のユーザーにも頒布してあげてください。いまのままでは片手落ちですよ。

浅野　憲(19)　X68000 PRO，X1turboIII，X1F，MZ-80C，FM-77L2，M5Jr.，PC-6001，PC-1245　大阪府

## ごめんなさいのコーナー

**6月号　よいこのSX-WINDOW講座**

P.123　記事中で，C言語のデータ型定義の記述に誤りがありました。以下のように変更してください。

```
typedef struct dlgItem2 {
    long            dlgIHdl;
                :
    unsigned char dlgIData [32];
} dlgItem2;
```

**バグに関するお問い合わせは**
**☎03(5488)1311(直通)**
**月〜金曜日16：00〜18：00**

お問い合わせは原則として，本誌のバグ情報のみに限らせていただきます。入力法，操作法などはマニュアルをよくお読みください。

また，よくアドベンチャーゲームの解答を求めるお電話をいただきますが，本誌ではいっさいお答えできません。ご了承ください。

## 来月号より やむなく 値上げします

▼BASICにはいろいろな制約がつきまといますが，やはり，他言語にくらべたときの手軽さは無視できません。今回の特集「Personal Tool,BASIC」では，このBASICをどのような場面で，どのように使うかというあたりを考えてみました。初心者の方はもちろん，他言語をメインに使っている人も，用途によってはどんどんBASICを使ってみてはいかがでしょうか。なんといっても，はじめから本体に付属している唯一の言語ですからね。
▼特集と連動する形で，今月号には別冊付録「X-BASIC簡易リファレンスブック」をつけてみました。編集の際には，『電脳倶楽部』に掲載されたオンラインヘルプ用ファイルをベースにさせていただきました。この場を借りて感謝の意を表したいと思います。皆さんもぜひお手元に置いて役立ててください。
▼さて，この付録がついた関係で，今月もOh!Xは特別定価600円になりました。が，実はつぎの8月号からOh!Xは値上げして，定価600円になることが決定しています。皆さんの負担が増えることは心苦しいのですが，より一層誌面充実に取り組むつもりですので，なにとぞご容赦願います。
▼3月号の特集に「西川善司のデモテーププレゼント」というのがありました。応募していただいた方には長らくお待たせしてしまいましたが，ここに当選者を発表します。
川端　洋之，畑中　道人，畑　剛志（北海道），佐々木　拓雄，渡辺　司芳（埼玉県），進藤　慶到，入本　泰光（東京都），宮越　良幸（神奈川県），小林　智（兵庫県），中村　志郎（福岡県）
　以上，10名の皆さんにデモテープをお送りします。おめでとうございます。
▼今月は「AFTER REVIEW」，「シミュレーション入門」がお休みです。特に「シミュレーション入門」は先月休んでいることもあり，なんとしてもお届けしたかったのですが，作者多忙のため締め切りに間に合いませんでした。もうしわけありません。

**投稿応募要領**

●原稿には，住所・氏名・年齢・職業・連絡先電話番号・機種・使用言語・必要な周辺機器・マイコン歴を明記してください。
●プログラムを投稿される方は，詳しい内容の説明，利用法，できればフローチャート，変数表，メモリマップ（マシン語の場合）に，参考文献を明記し，プログラムをセーブしたテープ（ディスケット）を添えてお送りください。また，掲載にあたっては，編集上の都合により加筆修正させていただくことがありますのでご了承ください。
●ハードの製作などを投稿される方は，詳しい内容の説明のほかに回路図，部品表，できれば実体配線図も添えてください。編集室で検討のうえ，製作したハードが必要な場合はご連絡いたします。
●投稿者のモラルとして，他誌との二重投稿，他機種用プログラムを単に移植したものは固くお断りいたします。

**あて先**
〒108 東京都港区高輪2-19-13 NS高輪ビル
ソフトバンク出版部
Oh!X「㋢㋷㋮㊔」係

# SHIFT・BREAK

▶腕時計が壊れてしまった。時計を持たなくなってみると，電車を待つときなんかにイライラしなくなって気分的にもおおらかになった。こりゃ，精神衛生上いいと思って数カ月。最近は時間を守れなくなって他人のひんしゅくを買ってばかり。おおらかになったんじゃなくて，単にいい加減な野郎になっていただけなのだった。やっぱり時計買お。（浦）
▶最近，レポートに追われている。「理系だからしょうがない」と人はいう。はたしてそうだろうか。「しょうがない」ではすまされないほど，レポート中心の生活が続いている。日曜日も空き時間もレポート。みんなこうやって大人になっていくのだろうが，ちょっと抵抗してやりたくなった。今度の日曜日はどこかへ遊びにいこう。（S.K.）
▶春から大学のパソコン（PC-9801RX）を使った授業が始まった。最初の授業はフォーマットの説明から始まり，各自持ってきた10枚のディスクをフォーマットして終わった。フォーマット済みのディスクを買った人は空しかっただろう。見渡すとハードディスクの扱い方を知らずに，アクセス途中でも電源を落とす人間がいる。ああ，こわい。（H.K.）
▶PDSの「DI ver.0.5」は編集部内でもスタンダードな存在。欠点はファイルが200数ファイル以上になるとまともに動作しなくなること。巷には新しいDIがあるようだが，カスタマイズ手順の複雑さやサイズが大きいのが気に入られず，いまだに編集部では「0.5」なのである。だれかバグの取れた「DI 0.5」持ってませんか。（善）
▶DRIVE ONのコーナーは一筆者としてもなにかと参考にさせていただいています。新モニタの方々，期待していますので，また有用なご意見をお寄せください。で，編集さんにこっそりお願いがあります。安井さん，贔屓してもらえません？　結構，彼女のレポート楽しみにしているんですよ。載らない号があったら泣きますからね。（Mu）
▶つまるところ，酷使しすぎたのだと思う。最近，物忘れが激しくなった。記憶の糸を辿り損ねる。脳が損傷している。損傷した部分を回避すればとりあえず働くのであるが，そこに記憶されていた情報は失われる。不調を抱えつつも，だましだましそしらぬ顔で働き，寿命を縮めていく。真剣に延命策を考えなければならなくなってきたようだ。（K）
▶タコから連想するもの。たこハイ，ぎゅわんぶらあ自己中心派。ペンギンから連想するもの。生ビール，スイートメモリーズ。すべては大学時代の思い出だ。プレイしてふと懐かしい気持ちになるのはそのせいか。自機にタコがなかったら，敵にペンギンがいなかったら，これほど「パロディウスだ！」にのめり込んだか。疑問だ。（KO）
▶6月号の編集後記で周りからつつかれたおかげで，「しばらく潜伏していて皆を驚かそう」計画が見事崩された。パソコンフォーラムではX68000のセカンドバッグを紛失してしまうし，最近はろくなことがない。気持ちをあらたにがんばるぞ！　と，6月号のハガキを読んでいくと……近藤英二さん，小井田伸雄さん，岩瀬貴代美さん，ごめんなさい。（J）
▶少し前のことだが，とある自然公園に行ったときの話。その公園には檻の中に多数の鳥が飼われているのだが，その鳥たちは羽を半分ぐらい切られている。筋を1本ではなく，羽そのものを半分。鳥にとってはどちらも変わりはないし，切る側もいろいろと配慮があってのことだろうけど，やはり見る側にとっては気持ちいいものではなかった。（A）
▶さても6月，ときは満ちた。寝袋，ゲームボーイ，ウォークマン，よし，準備万端。おっと忘れちゃいけない体力作り。これをやらずして救急車で運ばれたライバルたちが何人いたことか。おいっちに，さんしぃ……え？　いったいなんのことかって？　決まってるじゃない，某アーティストのイベントチケット争奪戦よ。目指せ今年も10列目以内！（E.O.）
▶68000CPUを引っこ抜き，倍速クロックのCPU基板を差し込む。基板上のSRAMとキャッシュコントローラのおかげでオーバーヘッドは少ない。ソフトウェアでのクロック切り換えも可能だ。Prince of Persiaを走らせる。もう重くなることはない。やっぱりPC-9801版より動きがずっとよく思える。日本版のほうがデキがいいといったのは誰だ？（U）
▶今年のビジネスショウでは，解像度の高い大きなCRTの数がずいぶん多かった。もうWindows 3.0などは解像度が640×400じゃ情けない。SX-WINDOWはいい線いっているけど，1024×768くらいあると結構違う。いいディスプレイ（長残光かなんかの）出ないかなあ。ところで，5月12日に私の長話を聞いてくれた人，どうもありがとう。（T）

## microOdyssey

このあいだ，ひさしぶりにファミコンソフトを買ってしまった。ファミコンなどの家庭用ゲーム機器にはもはや燃えられなくなって，本体そのものを押し入れの奥にしまってあったにもかかわらずである。

買ったのは新しいソフトではなく，古くて売れないという理由で安売りされているソフトだ。こういうふうに，はるか昔に発売されたソフトを安くなった時点で買って楽しむという方法は大学生のときにはよくやっていた。「金はないけど暇はある」，そういう生活だったからこそできることである。

毎朝(正確にいうと毎日昼過ぎに)，新聞の折り込み広告に目を通し，めぼしいものがあれば，古いファミコン雑誌をひっぱりだしてきて，「買うべきか，買わざるべきか」を検討する。

こういう苦労の末，安くて面白いソフトに当たったときの感動はひとしおである。実際，値段がある程度安ければ，面白くなくてもあまり損をした気にはならないものだ。

もちろん，本当に面白そうなソフトは定価でもいいから，発売当日，あるいは直後に買う。いいものにはそれ相応の金を出す価値があるし，買うほうの義務でもあると思うからだ。

最新のいいソフトが安い値段で手に入る。それがいちばんいいのであるが，商売だからそうもいかない。しかし，せめて発売されて時間がたったものぐらいは，たとえ面白いものでもなるべく安く買いたい。そう思うのはごく自然な考え方ではないだろうか。

パソコンソフトの場合はどうだろう。もちろん，同様のことはあるだろう。最近では，「SOF BOX」とかいうシリーズが本屋さんで売られていて，昔の名作ゲームを安く買うこともできる。これはいい傾向だと思う。

ところで，海外のソフトでは次のようなシステムがある。新作ソフトを買って中を開けてみると，自社作品宣伝のチラシが入っていて，最新作はもちろん昔の名作ソフトも載っている。そして，そういう古いソフトはディスカウントされていることが多い。

ソフトハウス自身がこういうことをやるというシステムは日本にはあまりないようだ。もちろん，あちらの場合はソフトハウスが流通もやっているという例が少なくない，という事情も関係しているだろう。

違う業界ならこういうシステムが日本にも存在する。CDである。洋楽の昔のアーチストのアルバムなどは安価に売られていることが多い。これはレコード屋さんではなく，レコード会社自身がやっている。

理由としては，「安い輸入レコードに対抗するため」ということが大きいのだろうが，昔の作品に関しては「もう十分元をとったから」という考えも少なからずあるようだ。

パソコンソフトの場合にも，先ほどの「SOF BOX」のようなシステムが出てきて，とりあえずは今後が期待できるかたちにはなっているのだが，やはりソフトハウスの積極的な姿勢というのもほしい。ソフトハウス自身がやるほうがより入手しやすいだろうし，「いいソフト」には時間の経過とともに埋もれさせていくのではなく，より大勢の人に楽しんでもらいたいという気持ちがあるはずだから。　(A)

## 1991年8月号7月18日(木)発売

**特集 スキャナ&プリンタの活用**
〜あるいはDTPへの遥かなる道程
**X1用ゲーム DEFEAT 2**
*Oh! X LIVE in '91* ゲームミュージック特集
全機種共通システム
**Small-Cの移植〈ライブラリ編〉**
定価600円

## バックナンバー常備店

| | | |
|---|---|---|
| 東京 | 神保町 | 三省堂神田本店5F 03(3233)3312 |
| | 〃 | 書泉ブックマートB1 03(3294)0011 |
| | 〃 | 書泉グランデ5F 03(3295)0011 |
| | 秋葉原 | T-ZONE 7Fブックゾーン 03(3257)2660 |
| | 八重洲 | 八重洲ブックセンター3F 03(3281)1811 |
| | 新宿 | 紀伊国屋書店本店 03(3354)0131 |
| | 高田馬場 | 未来堂書店 03(3200)9185 |
| | 渋谷 | 大盛堂書店 03(3463)0511 |
| | 池袋 | リブロ池袋店 03(3981)0111 |
| | 〃 | 西武百貨店9F コンピュータ・フォーラム 03(3981)0111 |
| 神奈川 | 横浜 | 有隣堂横浜駅西口店 045(311)6265 |
| | 〃 | 有隣堂ルミネ店 045(453)0811 |
| | 藤沢 | 有隣堂藤沢店 0466(26)1411 |
| 神奈川 | 厚木 | 有隣堂厚木店 0462(23)4111 |
| | 平塚 | 文教堂四の宮店 0463(54)2880 |
| 千葉 | 柏 | 新星堂カルチェ5 0471(64)8551 |
| | 船橋 | リブロ船橋店 0474(25)0111 |
| | 〃 | 芳林堂書店津田沼店 0474(78)3737 |
| | 千葉 | 多田屋千葉セントラルプラザ店 0472(24)1333 |
| 埼玉 | 川越 | 黒田書店 0492(25)3138 |
| | 川口 | 岩渕書店 0482(52)2190 |
| 茨城 | 水戸 | 川又書店駅前店 0292(31)0102 |
| 大阪 | 北区 | 旭屋書店本店 06(313)1191 |
| | 都島区 | 駸々堂京橋店 06(353)2413 |
| 京都 | 中京区 | オーム社書店 075(221)0280 |
| 愛知 | 名古屋 | 三省堂名古屋店 052(562)0077 |
| | 〃 | パソコンΣ上前津店 052(251)8334 |
| | 刈谷 | 三洋堂書店刈谷店 0566(24)1134 |
| 長野 | 飯田 | 平安堂飯田店 0265(24)4545 |
| 北海道 | 室蘭 | 室蘭工業大学生協 0143(44)6060 |

## 定期購読のお知らせ

Oh!Xの定期購読をご希望の方は綴じ込みの振替用紙の「申込書」欄にある『新規』『継続』のいずれかに○をつけ，必要事項を明記のうえ，郵便局で購読料をお振り込みください。その際渡される半券は領収書になっていますので，大切に保管してください。なお，すでに定期購読をご利用の方には期限終了の少し前にご通知いたします。継続希望の方は，上記と同じ要領でお申し込みください。

**海外送付ご希望の方へ**

本誌の海外発送代理店，日本IPS(株)にお申し込みください。なお，購読料金は郵送方法，地域によって異なりますので，下記宛必ずお問い合わせください。

日本IPS株式会社
〒101 東京都千代田区飯田橋3-11-6
☎03(3238)0700


**Oh!X** 7月号
■1991年7月1日発行　特別定価600円(本体583円)
■**発行人** 孫　正義
■**編集人** 橋本五郎
■**発売元** ソフトバンク株式会社
■**出版事業部** 〒108 東京都港区高輪2-19-13　NS高輪ビル
Oh!X編集部 ☎03(5488)1309
出版営業部 ☎03(5488)1360 FAX 03(5488)1364
広告センター ☎03(3297)0181
■**印　刷** 凸版印刷株式会社

# BACK ISSUES バックナンバー案内

ここには1990年7月号から1991年6月号までをご紹介しました。現在1990年11，12，1991年1～6月号の在庫がございます。バックナンバーおよび定期購読の申し込み方法については，172ページを参照してください。

## 1990

### 7月号（品切れ）
特集　マシン語への第一歩
X68000SUPER-HD試用レポート
連載　ショートプロぱーてぃ/Z80's Bar/DōGA・CGA
X-BASIC調理実習/PurePASCAL
●INTEGRAL XI――ノーマルXIへの対応
●ハードウェア工作入門
LIVE in '90　夢幻戦士ヴァリスII/トッカータとフーガニ短調
THE SOFTOUCH　サーク/あーくしゅ/ダウンタウン熱血物語
全機種共通システム　リロケータブルアセンブラWZD

### 8月号（品切れ）
特集　ADVANCED 2D GRAPHICS
100号記念特別モニタプレゼント
連載　ショートプロぱーてぃ/Z80's Bar/INTEGRAL XI
X-BASIC調理実習/X68000マシン語プログラミング
PurePASCAL/ハードウェア工作入門
●X68000用画像回転プログラム　XROT0.X
LIVE in '90　OMENS OF LOVE/ENDLESS RAIN/ダートフォックス
THE SOFTOUCH　大航海時代/ウルティマV/プロミストランド
全機種共通システム　リンカWLK

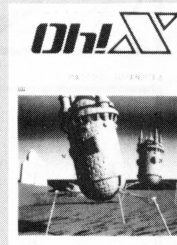
### 9月号（品切れ）
特集1　日本語を処理するための序章
特集2　ADVANCED 2D GRAPHICS
連載　ショートプロぱーてぃ/Z80's Bar/DōGA・CGA
X-BASIC調理実習/マシン語プログラミング
PurePASCAL/ハードウェア工作入門
●清水和人流プログラミング道場
LIVE in '90　風の谷のナウシカ/ラジオ体操第一
THE SOFTOUCH　T&T/D-Again/シムシティー/ギャラガ'88ほか
全機種共通システム　BILLIARDS

### 10月号（品切れ）
特集　電子音楽術入門
連載　ショートプロぱーてぃ/Z80's Bar/DōGA・CGA
マシン語プログラミング/ハードウェア工作入門
清水和人流プログラミング道場
●荻窪圭の大人のためのX68000
●中森章のようこそここへC言語
LIVE in '90　Rise And Fall/PARADOX/キューピー3分クッキング
THE SOFTOUCH　ワールドコート ルーンワース 闇の血族 提督の決断
全機種共通システム　ライブラリアンWLB

### 11月号
特集　理科系のGAME REVIEW
連載　Z80's Bar/DōGA・CGA/カードゲーム
マシン語プログラミング/ハードウェア工作入門
PurePASCAL/X-BASIC調理実習
ようこそここへC言語/INTEGRAL XI
荻窪圭の大人のためのX68000
LIVE in '90　ピラミッドソーサリアン/ザ・スキーム
THE SOFTOUCH SPECIAL　ラグーン/幻獣鬼/サイバリオン/GUNSHIP他
全機種共通システム　スクリーンエディタEDC-T

### 12月号
特集　XCのための傾向と対策
連載　X-BASICプログラミング調理実習/ハードウェア工作入門
マシン語プログラミング/ショートプロぱーてぃ/Z80's Bar
大人のためのX68000/ようこそここへC言語/INTEGRAL XI
●シミュレーションプログラミング入門
●特別企画アナログジョイスティックの製作
LIVE in '90　グラディウスIII/メタルサイト
THE SOFTOUCH SPECIAL　イメージファイト/ジェミニウイング/NAIOUS他
全機種共通システム　STACKコンパイラ

## 1991

### 1月号
特集　急接近！SX-WINDOW
特別付録　謹賀新年PRO-68K（5"2HD）
連載　ハードウェア工作入門/シミュレーションプログラミング入門
DōGA・CGA/ショートプロぱーてぃ/大人のためのX68000
PurePASCAL/清水和人流プログラミング道場/X-BASIC調理実習
LIVE in '91　めぞん一刻/涙で綴るパパへの手紙
THE SOFTOUCH　ソル・フィース/銀英伝II/続ダンジョン・マスター他
製品紹介　光磁気ディスクCZ-6MO1
全機種共通システム　ライブラリアンWLB

### 2月号
特集1　グラフィックの“実験的”手法
特集2　SX-WINDOWプログラミング
連載　ハードウェア工作入門/シミュレーションプログラミング入門
マシン語プログラミング/大人のためのX68000/Z80's Bar
ショートプロぱーてぃ/INTEGRAL XI/ようこそここへC言語
●1990年度 GAME OF THE YEAR ノミネート発表
LIVE in '91　Misty Blue/スプーンおばさん
THE SOFTOUCH　栄冠は君に/KLAX/ダイナマイト・デューク他
全機種共通システム　ダイスゲームKISMET

### 3月号
特集　MIDI & MUSIC PROCESSING
連載　ハードウェア工作入門/シミュレーションプログラミング入門
マシン語プログラミング/大人のためのX68000/Z80's Bar
ショートプロぱーてぃ/DōGA・CGA/C言語/PurePASCAL
●SXLIFE完結編/ウィンドウシステム大比較
●周辺機器新製品紹介
LIVE in '91　戦いの兜/LITTLE WING/リゾ・ラバ/花
THE SOFTOUCH　アトミック・ロボキッド/スペースローグ他
全機種共通システム　アクションゲームMUD BALLIN'

### 4月号
特集　人とゲームのインタフェイス
連載　DōGA・CGA/シミュレーションプログラミング入門
ハードウェア工作入門/ようこそここへC言語/Z80's Bar
ショートプロぱーてぃ/清水和人流プログラミング道場
●新連載　吾輩はX68000である/よいこのSX-WINDOW講座
●決定！1990年度GAME OF THE YEAR
LIVE in '91　Easy Come, Easy Go!/シシリエンヌ
THE SOFTOUCH　メルヘンメイズ/中華大仙/スライス他
全機種共通システム　SLANG用カードゲームDOBON

### 5月号
特集　新登場！　X68000XVI/XVI-HD
特別付録　黄金週間PRO-68K（5"2HD）
第6回　言わせてくれなくちゃだワ
連載　ハードウェア工作入門/ようこそここへC言語
大人のためのX68000/X68000マシン語プログラミング
ショートプロぱーてぃ/マシン語カクテルin Z80's Bar
LIVE in '91　ブービーキッズ/NO. NEW YORK
THE SOFTOUCH　マーブル・マッドネス/シグナトリー/石道他
全機種共通システム　実数型コンパイラ言語REAL

### 6月号
特集　初心者のための環境構成術
創刊9周年記念Oh! Xアンケート結果大分析大会その1
連載　ハード工作/大人のためのX68000/Z80's Bar/DōGA・CGA
ようこそここへC言語/ショートプロぱーてぃ/よいこのSX-WINDOW講座
吾輩はX68000である/マシン語プログラミング
●響子 in CGわーるど●マウスアダプタの製作
LIVE in '91　暴れん坊将軍/ナディア/POWER HALL他
THE SOFTOUCH　パロディウスだ！/遥かなるオーガスタ/ノスタルジア他
全機種共通システム　S-OS 6周年記念Small-C処理系の移植

# 満開の電子(でんこ)ちゃん

作：いわいいっぺい

え：岡村祭

7月7日
星印乳業㈱

ここでその牛が笑ったそうだ
うっしっし
なーんてな
……

チーズ10グロス配送にまわりました

牛乳鮮度OKです

ひこ星さんもうあがっていいですよ

そうそう年に一回のデートでしょ

それが…さっきからバターが見あたらなくて

えっあの大口の？

HOSHIZIRUSI

ああっ!!待ち合わせがバターのおり姫で納期が

あわてないでひこ星さんそんな時はこれよっ!!

あっそれは電脳倶楽部!!

電脳倶楽部はX68000のためのディスクマガジンなのいろんなツールや楽しい音やビープ音役立つデータがいっぱい入っているのよ

しかも電源オンでたちまち起動あとはマウスでラクラク操作なの

ぽんっ!

そうだっ落ち着いたら思い出したぞ

あの棚だっ

星印☆バター

これがホントの棚(タナ)バター!!

バッタがいれば棚バッタ

ホームランだよ棚バッター!

無理矢理すぎるぞ棚バルタン!

その頃

天の川駅
東口

ざわざわ

ガヤガヤ

いこいの温泉

伝言板
ひこ星へ
さんざん待ったけど
帰るわ
おり姫



平山 誠（北海道）

「電脳倶楽部」――もう知らない人はいませんね。すでに多くの仲間が購読しています。システムインストゥールも、難しい操作も不要。すぐ役に立つ、便利なツールやグラフィックが満載なのですから、見逃す手はありません。おまけに、お友達を紹介すると編集長のクリスタル像が……という事もありませんから、このモテモテぶりは当たり前、編集部には購読者からの「電脳倶楽部を知ってから明るくなりました」などのお礼の手紙が山積みだそうです（うそ50%）。さあ、君も今すぐ郵便局へGO!

# 業　務　連　絡

## 赤えんぴつ（JRA版）

お客様各位へ

平素は、当社の製品をご愛顧いただき、誠に有難うございます。
この度、『赤えんぴつ』用の'91年度版の『データ・ディスク』が出来上がりました。
この'91年版の『データ・ディスク』は旧データ・ディスクに'90年の各競馬場のデータを追加し、予想をする時の基本データを修正して当たる確率をアップしました。
愛用者カードを頂いたお客様には、案内状をお送り致しましたが、『赤えんぴつ』に添付の愛用者カードをまだ当社へ返送されていない方は、お手数ですがお早めにご返送していただく様お願い致します。
また、同時にお客様からのご要望の多かった下記の点を改良した、『赤えんぴつ』のバージョンアップ版を制作致しました。
これは、'91年版の『データ・ディスク』をお申込頂いた方に無料で同梱いたします。

Ⅰ. カーソルコントロールキーだけで行っていたデータの入力の殆どを、テンキーからの直接入力で出来るように成りました。
Ⅱ. キーバッファーを切って、カーソルコントロールキーを幾ら押しても指を離せばそこからは入力されなくなりました。
Ⅲ. カーソルコントロールキーでデータ入力する時にデータの終端を無くしたエンドレス・スクロールにしました。
Ⅳ. 今迄は、予想の結果をCRTのみに出力していたのを、プリンターへのプリントアウトが可能に成りました。
Ⅴ. パーソナル・ディスクからデータを読み込んだ後、セーブしないで処理を終了すると自動的にデータが消去されていたのが、消去するか否かを確認してから消去する様にしました。

**【有料モニターの募集】**
『赤えんぴつ』を一度試してみたい方に4,000円で一部の機能（データのロード・セーブ、結果のプリントアウト等）を除いた簡易型の『赤えんぴつ』をお送り致します。
お試し頂いた上で本来の『赤えんぴつ』をご希望の方に16,000円でお求めになれます。
なお、この有料モニター・サービスは当社へ直接お申込み下さい。

**赤えんぴつ**　X68000用　2HD　**20,000円**

＊商品の価格には消費税は含まれていません。

▶ **お求めは全国の有名マイコンショップでどうぞ。**
通信販売をご希望の方は当社へ直接、商品名・機種名・メディア名・住所・氏名・電話番号を明記の上、現金書留にてお申し込みください。（送料無料）

BLUE SKY Co.
株式会社　BLUE SKY
〒411　静岡県三島市加茂16-4　☎0559-72-6710

# 最大メモリ8Mバイト

# KGB-X68PRKII

新発売!

- 8M増設メモリと数値演算プロセッサが1枚のボードに収まります。
- 従来品(KGB-X68PRK)に比べて大幅なコストダウン。
- メモリ容量2M/4M/6M/8Mの4種類、それぞれに数値演算プロセッサ有無のモデルを用意しました。
- 数値演算プロセッサ無しモデルではMC68881RC16の購入で簡単にグレードアップが可能です。
- 当然、2M/4M/6Mモデルでは購入後も8Mまでのメモリ増設が可能です。
- XVI用の専用内蔵RAMとの共存はできません。

| 品名 | 型番 | 価格 |
|---|---|---|
| 2Mメモリ数値演算プロセッサ無し | PRKII-02 | ¥ 55,000 |
| 4Mメモリ数値演算プロセッサ無し | PRKII-04 | ¥ 90,000 |
| 6Mメモリ数値演算プロセッサ無し | PRKII-06 | ¥ 125,000 |
| 8Mメモリ数値演算プロセッサ無し | PRKII-08 | ¥ 160,000 |
| 2Mメモリ数値演算プロセッサ付属 | PRKII-12 | ¥ 85,000 |
| 4Mメモリ数値演算プロセッサ付属 | PRKII-14 | ¥ 120,000 |
| 6Mメモリ数値演算プロセッサ付属 | PRKII-16 | ¥ 155,000 |
| 8Mメモリ数値演算プロセッサ付属 | PRKII-18 | ¥ 190,000 |

## PRKII質問箱

Q 購入後のメモリ増設はどうやるのでしょう?
A ご購入後のPRKIIに対するメモリも増設は半田付けなどの技術を要するためボードを当社に送り返していただき増設をいたします。ご自分でメモリ増設をする場合には部品の販売も予定しております。

Q 数値演算プロセッサにMC68882を使用することは可能ですか?
A MC68882では動作しないソフトが存在するために使用することは出来ません。

Q 旧PRKとPRKIIではどこが違うのですか?
A 1枚に収まるメモリが最大で8Mになった以外は同じです。

Q 数値演算プロセッサを使うと速度が速くなるのですか?
A 数値演算プロセッサを使用することにより速くなるのは実数演算のみです。画面表示などは速くなりません。

## 充実のBASIC HOUSEソフトウェア&ハードウェア

| 品名 | 機種 | 価格 |
|---|---|---|
| 高速12BIT, 16CH A/Dコンバータボード(KGB-AD12) | X1 | ¥118,000 |
| フォトアイソレーション16BITデジタル入出力ボード(KGB-PIO) | X1 | ¥ 42,000 |
| アイソレーション16BITデジタル入出力ボード(KGB-X68PIO) | X68000 | ¥ 68,000 |
| ハンディプリンタ&インターフェース(HANDYPRINTjack) | X68000 | ¥ 24,800 |
| 高速12BIT, 4CH D/Aコンバータボード(KGB-DA4) | X1 | ¥ 98,000 |
| 汎用ローコストA/D&PIOボード(KGB-X1S) | X1 | ¥ 19,800 |
| 高速12BIT, 16CH A/Dコンバータ(KGB-X68ADC) | X68000 | ¥128,000 |
| 64180CPUボードMach 180(KGB-CPXB) | X68000 | ¥ 98,000 |
| ローコストMIDIインターフェース(MELODY BOX) | X68000 | ¥ 16,800 |

- BASIC拡張関数パッケージ(B6-6301) ¥9,800
- C言語ライブラリ(B6-6305) ¥6,800
- ディスクキャッシャー(B6-6304-2) ¥6,800
- Toys & Tools (B6-6307) ¥6,800
- BASIC拡張関数パッケージC言語ライブラリ付(B6-6306) ¥14,800
- アイコンエディタ(B6-6303) ¥4,800
- CP/M68Kエミュレータ(B6-6302) ¥19,800

## おしらせ

### スタッフ募集!

計測技研/First Class Technologyでは、プログラマースタッフを募集しています。

X68000大好き人間、新しい物好きの明るい人、いっしょに開発しましょう。

ご希望の方は、計測技研高橋までご連絡下さい。

ビデオボードを外付けに!!
ビデオボードケース(KGB-BVBX)
**大好評発売中 定価9,800円**

SHARPより発売されているCZ-6BV1を外付けにするケースです。このケースの使用によりあなたのX68000のスロットが開放されます。

Human68k下のソフトのCRT出力を強制的に15kHZ出力にする(768×512モード除く)おまけユーティリティ付き

表示されている価格は消費税別の価格です。

全国どこでも発送可 長期クレジットOK 送料全国均一¥1,000 宅配便にて即日配送

株式会社計測技研
本社営業部/マイコンショップ/通販部 宇都宮市竹林町503-1 TEL0286-22-9811 FAX0286-25-3970
大田原営業所/マイコンショップ 大田原市美原1-13-4 TEL0287-23-5352 FAX0286-23-5364
マイコンショップ BASIC HOUSE お申し込み・お問い合せは ☎0286-22-9811(代)

# AIBIT
アイビット電子株式会社

# SHARP
## パソコン本体から周辺機器まで品数取り揃え
## 大特価セール実施中!!

| 型名 | 品名 | 正価 | 特価 |
|---|---|---|---|
| PC-E500BL | ポケコン | 28,800 | **19,500** |
| PC-1600K | ポケコン | 69,800 | **49,800** |
| PC-1360K | ポケコン | 36,800 | **32,800** |
| PC-1360 | ポケコン | 29,800 | **19,800** |
| PC-1262 | ポケコン | 24,800 | **19,600** |
| PC-1248DB | ポケコン | 11,000 | **9,800** |
| PC-1280 | ポケコン | 24,800 | **19,600** |
| CE-T800 | ポケコンRS-232Cコンバーター | 12,800 | **11,800** |
| CE-203M | ポケコンRAM32K | 32,000 | **7,000** |
| CE-202M | ポケコンRAM16K | 35,000 | **6,000** |
| CE-201M | ポケコンRAM 8K | 18,000 | **3,000** |
| CE-1600M | ポケコンRAM32K | 32,000 | **16,000** |
| CE-1600F | ポケコンフロッピードライブ | 39,800 | **34,800** |
| CE-1600P | ポケコンプリンター | 69,800 | **59,800** |
| CE-1650F | ポケコンDISK | 9,800 | **8,800** |
| CE-161 | ポケコンRAM16K | 50,000 | **3,800** |
| CE-1601M | ポケコンRAM64K | 45,000 | **30,000** |
| CE-1600E | ポケコンディスクインターフェイス | 19,800 | **17,800** |
| CE-158 | ポケコンレベルコンバター | 39,800 | **31,300** |
| CE-159 | ポケコンRAM 8K | 35,000 | **4,200** |
| CE-140T | ポケコンRS-232Cコンバター | 9,800 | **8,800** |
| CE-140F | ポケコンフロッピディスク | 49,800 | **44,800** |
| CE-123P | ポケコンプリンター | 19,800 | **17,800** |
| CE-120P | ポケコンプリンター | 24,800 | **21,800** |
| CE-126P | ポケコンプリンター | 17,800 | **13,800** |
| CE-124 | ポケコンカセットインター | 4,500 | **3,600** |
| Z-VISIONplus | Z80シュミレータ デバッカー | 59,800 | **51,000** |
| UX-1 | ホームコピーファクス | 78,000 | **69,800** |
| PA-9500 | ハイパー電子手帳 | 48,000 | **特価** |
| CZ-300F | X13"マイクロフロッピー | 79,800 | **9,000** |
| CZ-31FS | 300F増設フロッピー | 59,800 | **7,000** |
| CZ-82F | CZ-802C増設フロッピー | 59,800 | **6,000** |
| CZ-501H | X1増設用ハードディスクユニット | 258,000 | **60,000** |
| CZ-6BS1 | SCSIボード | 29,800 | **23,800** |
| CZ-6BP1 | 数値演算ボード | 79,800 | **63,800** |
| CZ-6BU1 | ユニバーサルI/Oボード | 39,800 | **33,800** |
| CZ-6BM1 | MIDIボード | 29,800 | **23,800** |
| CZ-6BEIB | 1M増設RAMボード | 28,000 | **19,500** |
| CZ-6BE1 | 1M増設RAMボード | 35,000 | **29,500** |
| CZ-6BE2 | 2M増設RAMボード | 79,800 | **63,800** |
| CZ-6BE4 | 4M増設RAMボード | 138,000 | **110,400** |
| CZ-6BN1 | スキャナーボード | 29,800 | **25,300** |
| CZ-6BF1 | RS-232C増設ボード | 49,800 | **42,300** |
| CZ-6SD1 | システムラック | 44,800 | **38,000** |
| CZ-6TU | RRGBシステムチューナー | 33,100 | **26,500** |
| CZ-6BG1 | X6800GPIBボード | 59,800 | **50,000** |
| CZ-6BC1 | X6800FAXボード | 79,800 | **67,800** |
| CZ-6PV1 | ビデオプリンター | 198,000 | **158,000** |
| CZ-6BV1 | ビデオボード | 21,000 | **16,800** |
| CZ-822C | X1G MODEL30 | 118,000 | **39,800** |
| CZ-820C | X1G MODEL10 | 69,800 | **16,800** |
| CZ-8BGR2 | グラフィックボードX1 | 14,800 | **3,000** |
| CZ-8BF1 | FDインターフェイス | 14,800 | **11,500** |
| CZ-8BK2 | X1 漢字ROM | 19,800 | **16,800** |
| CZ-8BM2 | 232Cマウスボード | 19,800 | **16,800** |
| CZ-8BE2 | 320K外部メモリー | 29,800 | **25,300** |
| CZ-8BR1 | 立体映像セット | 29,800 | **25,300** |
| CZ-8BV2 | カラーイメージボート | 39,800 | **32,000** |
| CZ-8BO1 | FDインターフェイス | 14,800 | **8,000** |
| CZ-8TM1 | X1ソフト付モデムユニット | 29,800 | **5,000** |
| CZ-8TM2 | X1ソフト付モデムユニット | 49,800 | **39,800** |

| 型名 | 品名 | 正価 | 特価 |
|---|---|---|---|
| CZ-8EB3 | 拡張i/o box | 33,800 | **28,000** |
| CZ-8LM1 | 232cケーブル | 7,200 | **6,000** |
| CZ-8LM2 | 232cクロスケーブル | 7,200 | **6,000** |
| CZ-8NJ1 | ジョイカード | 1,700 | **1,360** |
| CZ-8NT1 | トラックボール | 13,800 | **11,500** |
| CZ-8PK10 | 24ドット136桁漢字プリンター | 99,800 | **大特価** |
| CZ-8PK7 | 24ドット80桁漢字プリンター | 122,000 | **59,800** |
| CZ-8PC5BK | 48ドット熱転写カラー漢字プリンター | 96,800 | **新発売** |
| CZ-8BS1 | X1FM音源ボード | 23,800 | **19,800** |
| CZ-8BK4 | X1第2水準ROM | ——— | **5,700** |
| CZ-8NJ2 | インテリジェントコントローラー | 23,800 | **18,500** |
| CZ-8NS1 | カラーイメージスキャナー | 188,000 | **149,000** |
| AN-S100 | アンプ付スピーカー | 36,600 | **29,500** |
| AN-X68 | キーボードシリコンカバー | 3,500 | **2,800** |
| AN-X68PRO | キーボードシリコンカバー | 3,500 | **2,800** |
| AN-1508 | ディスプレイ15P→8P変換ケーブル | ——— | **1,600** |
| AN-1506 | ディスプレイ15P→6P変換ケーブル | ——— | **1,600** |
| HXD040 | アイティム40Mハードディスク(ITM) | 118,000 | **89,000** |
| HXD140 | 40Mハードディスク内蔵用(ITM) | 98,000 | **79,800** |
| CU-14FD | カラーディスプレーアナログ0.31 | 74,800 | **49,800** |
| MZ-1D10 | 12"モノクロディスプレー | 41,800 | **25,000** |
| MZ-1D17 | 15"CRT mz-5500/6500/2 | 124,000 | **59,800** |
| MZ-1E05 | MZ-2000 FDインターフェイス | 24,500 | **18,000** |
| MZ-1E08 | プリンターI/F 2000/2200/80B | 9,000 | **8,000** |
| MZ-1E11 | MZ-6500用 SFD I/F | 38,000 | **25,000** |
| MZ-1E04 | MZ-2000 プリンターI/F | 10,000 | **6,000** |
| MZ-1E21 | MZ-5500 GP I/F | 36,000 | **12,000** |
| MZ-1E18 | MZ2000QD用インターフェイス | 9,800 | **3,000** |
| MZ-1E33 | MZ6500パラレルI/F | 34,800 | **28,000** |
| MZ-1E45 | MZ6500 232C I/F | 50,000 | **15,000** |
| MZ-1E32 | MZ2500 パラレル I/F | 30,000 | **27,000** |
| MZ-1E44 | MZ-6500 S-RN I/F | 50,000 | **15,000** |
| MZ-1E22 | MZ-5500 GPIB I/F | 72,800 | **25,000** |
| MZ-1E29 | RS-232Cインターフェイス 300BT | 17,800 | **9,800** |
| MZ-1E01 | MZ-3500 232Cボード | 28,000 | **13,000** |
| MZ-1E14 | MZ1500 QD用インターフェイス | 9,800 | **3,000** |
| MZ-1M01 | MZ-2000/2200 16ビットボード | 78,000 | **8,000** |
| MZ-1M09 | MZ-6500 8082-2演算プロセッサ | 82,000 | **30,000** |
| MZ-1M03 | MZ-5500 数値演算 | 69,000 | **38,500** |
| MZ-1M12 | MZ-2861 8087 演算プロセッサ | 90,000 | **45,000** |
| MZ-80P4B | 136桁ドットプリンター | ——— | **48,000** |
| MZ-1P06 | ドットプリンター | 234,000 | **45,000** |
| MZ-1P27 | 水平漢字プリンター | 268,000 | **188,000** |
| MZ-1P28 | ドットプリンター漢字80桁 | 148,000 | **118,400** |
| MZ-1P10A | 24ドットプリンター漢字80桁 | 245,000 | **79,000** |
| MZ-1P22 | 熱転写漢字プリンター | 59,800 | **25,000** |
| MZ-1P29 | 漢字プリンター136桁 | 168,000 | **134,400** |
| MZ-1P30 | 136桁プリンター | 228,000 | **120,000** |
| MZ-1R01 | MZ-2000/2200Gボード | 39,800 | **10,000** |
| MZ-1R10 | MZ-5500 漢字ROM付 | 30,000 | **9,800** |
| MZ-1R09 | MZ-5500 V.RAM | 35,000 | **15,000** |
| MZ-1R06 | MZ-5500 増設RAM | 45,000 | **8,000** |
| MZ-1R12 | MZ-80B/2000/1500/700 RAM | 35,000 | **8,000** |
| MZ-1R11 | MZ-5500 256KRAM | 80,000 | **35,000** |
| MZ-1R36 | MZ-28611M増設RAM | 45,000 | **15,000** |
| MZ-1R35 | MZ-28611M増設RAM | 55,000 | **19,000** |
| MZ-1R14 | MZ-5500 辞書ROM | 40,000 | **22,000** |
| MZ-1R16 | MZ-5500 128KRAM | 30,000 | **8,000** |
| MZ-1R27A | MZ-2500VRAM | 13,000 | **10,000** |
| MZ-1R21 | 漢字ROM | 38,000 | **13,000** |
| MZ-1R24 | MZ-1500 辞書ROM | 22,000 | **6,000** |

| 型名 | 品名 | 正価 | 特価 |
|---|---|---|---|
| MZ-1R32 | MZ-6500RAM | 80,000 | **40,000** |
| MZ-1R31 | 漢字ROM | 28,000 | **20,000** |
| MZ-1R28A | MZ-2500 辞書ROM | 13,000 | **10,000** |
| MZ-1R29A | MZ-1P22 第2水準漢字ROM | 15,000 | **12,000** |
| MZ-1S13 | MZ-1D17チルトスタンド | 12,000 | **5,000** |
| MZ-1T02 | MZ-2200 データーレコーダー | 19,800 | **8,500** |
| MZ-1T03 | MZ-5500 データーレコーダー | 12,000 | **8,500** |
| MZ-1U09 | MZ-2500 拡張ボード | **代品在庫少々有り** | |
| MZ-1V01 | パソコン FAX | 278,000 | **85,000** |
| MZ-1X22 | モデムユニット | 21,800 | **13,000** |
| MZ-2Z016 | MZ-5500 附属 | ——— | **5,000** |
| MZ-2Z023 | MZ-5500 GWBASIC | 50,000 | **30,000** |
| MZ-2Z031 | MZ-6500 日本語ワープロ | 49,800 | **15,000** |
| MZ-2Z029 | MZ-6500 TODAY | 68,000 | **20,000** |
| MZ-2Z064 | MZ-6500 書院RAM付 | 69,800 | **28,000** |
| MZ-2Z065 | MZ-6500 書院RAMなし | 49,800 | **15,000** |
| MZ-2Z012 | MZ-5500 附属 | ——— | **5,000** |
| MZ-2Z013 | MZ-5500 MSDOS | 25,000 | **20,000** |
| MZ-4Z001 | MZ-5500 IBM変換 | 30,000 | **8,000** |
| MZ-5521 | 本体 | 388,000 | **55,000** |
| MZ-5511 | 本体 | 288,000 | **35,000** |
| MZ-5Z013 | MZ-1500 QD通信ソフト | ——— | **3,500** |
| MZ-6F03 | ブランク QD DISK | 450 | **400** |
| MZ-6P18 | MZ-1P18、28カットシートフィーダー | 60,000 | **35,000** |
| MZ-6P29 | MZ-1P29 カットシートフィーダー | 50,000 | **37,500** |
| MZ-6P27 | MZ-1P27 カットシートフィーダー | 58,000 | **39,800** |
| MZ-6P06 | MZ-1P06 トラクターフィード | 15,000 | **7,500** |
| MZ-6P20 | MZ-1P22/17ロールホルダー | 3,100 | **2,700** |
| MZ-6Z22 | MZ-6500(50)CP/M86BASIC-3 | 10,000 | **6,000** |
| MZ-6Z25 | M-50 ストリーマユーティリティZプロセッサ | 39,800 | **15,000** |
| MZ-80T20A | MZ-80マシンランゲージ | 6,000 | **5,000** |
| MZ-80TUB | MZ-80 バックアップ | 20,000 | **8,000** |
| MZ-80TU | MZ-80 システムプログラム | 20,000 | **8,000** |
| MZ-80T40A | MZ-80 PASCAL | 10,000 | **5,000** |
| MZ-80T70A | MZ-80 FDOS | 20,000 | **7,000** |
| MZ-8BGK | MZ-80 BGRAM2 | 39,000 | **10,000** |
| MZ-8B104 | MZ200/2200 GP IBインターフェイス | 45,000 | **18,000** |
| MZ-8BC01 | MZ200/2200 GP IBケーブル | 18,000 | **8,000** |
| UE-1U01 | X286L スロットBOX | 5,000 | **4,000** |
| UE-1R02 | 4M RAMボード | 300,000 | **240,000** |
| UE-1R06 | 辞書ROMボート | 32,800 | **25,600** |
| UE-1R01 | 2M RAMボード | 160,000 | **128,000** |
| UE-1R05 | 拡張グラフィックボード | 92,000 | **55,000** |
| UE-1R03 | 1M RAMボード | 100,000 | **80,000** |
| UE-1R04 | 2M RAMボード | 180,000 | **144,000** |
| UE-1P03 | 80桁漢字プリンタ | ——— | **特価** |
| UE-1P04 | 136桁漢字プリンタ | ——— | **特価** |
| UE-1P05 | 136桁漢字水平プリンタ | ——— | **特価** |
| UE-1P02 | 高速136桁漢字プリンタ | 550,000 | **440,000** |
| UE-1P01 | 136桁漢字プリンタ | 268,000 | **214,400** |
| UE-1E04 | S-RNインターフェイスカード | 70,000 | **56,000** |
| UE-1E02 | AX286L ICカードI | 45,000 | **36,000** |
| UE-1E03 | 5"FDインターフェイスカード | 28,000 | **22,400** |
| UE-1D03 | 15インチカラーディスプレイ | 123,000 | **98,400** |
| UE-1D02 | 14インチカラーディスプレイ | 158,000 | **126,400** |
| IO-735X | カラープリンター | 248,000 | **180,000** |
| BF-68PRO | フィルター | 19,800 | **16,800** |
| X68000キーボード延長ケーブル(1.5m) | | 2,500 | **2,000** |
| ディスプレーケーブルアナログ15P(3m) | | 5,000 | **4,000** |
| ディスプレーケーブルアナログ15P(1.5m) | | 4,300 | **3,500** |

**ポケコン関係周辺機器サプライ製品及シャープ関係のソフトウエア全種取扱います。**

**FM TOWNS/FM NOTE/東芝ダイナブック、周辺機器も取扱っております。**

# 夏のボーナス特価セール!!

シャープ パソコンフォーラム'91 **展示品大特価！** 台数限定・いまがチャンス

XVI CZ-634CTN
+CZ-606DTN 特価¥320,000
XVI CZ-644CTN
+CZ-606DTN 特価¥430,000

'91年7月15日迄

**X68000F下取りセール実施中！**

# AIBIT
アイビット電子株式会社

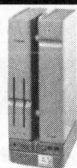
## SHARP X68000シリーズ対応 ハードディスク

〈ITEM〉

**HXD 040 23mS X68000**
定価¥118,000⇒特価¥79,800

**HXD 042 X68000 増設用**
定価¥128,000⇒特価¥102,500

**HXD 140 X68000 内蔵用**
定価¥98,000⇒特価¥79,800

HXD-140は602C、603Cの内蔵用

## X68000 お買上げの方に ニュージーランド(ソフト)¥8,800 + マウスパット¥3,000 + フロッピータイトルシール¥600 **3点セット** プレゼント(7/15まで)

### CZ-602CGY(本体)
プラス(ディスプレイ)組合せ

CZ-603DGY **¥248,000**
CZ-612DGY **¥268,000**

### CZ-602CGY(本体)
プラス(ディスプレイ+40MB HD)

CZ-603DGY **¥315,000**
CZ-612DGY **¥335,000**
CZ-613DGY **¥355,000**

### CZ-653CBK(本体)
プラス(ディスプレイ)組合せ

シャープ パソコンフォーラム'91 展示品(台数限定)
基本セット(CZ-606DBK) **¥228,000**

### CZ-604CTN(本体)
プラス(ディスプレイ)組合せ

CZ-606DTN **¥290,000**
CZ-613DTN **¥330,000**

### CZ-603C(本体)
プラス(ディスプレイ)組合せ

CZ-606D **¥270,000**
CZ-604D **¥280,000**
CZ-605D **¥300,000**
CZ-613D **¥310,000**

### CZ-623CTN(本体)
プラス(ディスプレイ)組合せ

CZ-606DTN **¥400,000**
CZ-613DTN **¥440,000**

## X1ソフト・X68000ソフト

### X1(turbo用)

信長の野望(戦国群雄伝)……¥ 8,330
三国志II……¥12,580
ランペルール……¥ 8,330
大航海時代……¥ 8,330
蒼き狼と白き牝鹿……¥ 8,330
水滸伝……¥ 8,330
ソーサリアン……¥ 8,330
ユーティリティ、追加シナリオ、戦国ピラミッドソーサリアン……各¥ 3,230
イースI・II……各¥ 6,630
マイト&マジックII……¥ 8,330

### X1(turbo不可)

ファンタジーIII……¥ 8,330
ザナドゥI……¥ 6,630
ザナドゥII……¥ 4,930

### X1用

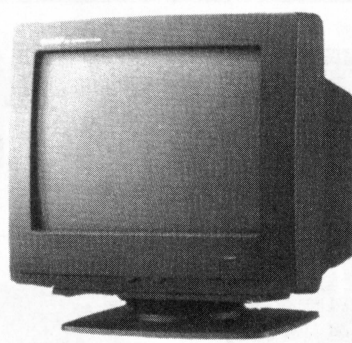
JETターボターミナル……¥ 8,330
MR-ASM-MR-ID……¥10,880
モデムターミナル……¥ 2,000
turbo CP/M……¥12,500
C……¥11,700
FōRTRAN……¥11,700
APL……¥11,700
X1 LōGō……¥ 8,330
X1turbo LōGō……¥15,980
Samurai ワープロ侍……¥16,800
Shōgun 2D ワープロ将軍……¥16,800
Shōgun 2HD ワープロ将軍……¥16,800

### X68000用

BUSINESSPRō68K CZ-212BS……¥57,800
CANVAS CZ-249GS……¥25,330
XBAS to C CHECKER CZ-260LS……¥ 8,330
Multiword CZ-225BS……¥27,200
Human68K Ver.20 CZ-244SS……¥ 8,330
Stationarg CZ-240BS……¥12,580
SōUND PRō68K CZ-214BS……¥13,430
Sampling PRō68K CZ-215MS……¥15,130
MUSIC PRō68K CZ-213MS……¥15,980
MUSIC PRō68K MIDI CZ-249MS……¥24,480
MUSIC Studiō PRō68K Ver.2.0 CZ-261MS……¥24,480
CARD PRō68K Ver.2.0 CZ-253BS……¥25,330
CARD 活用フォーム集 CZ-242BS……¥ 8,330
SX-WINDōW Ver.1.1 CZ-278SS……¥ 8,330
OS-9 CZ-219SS……¥25,330

## シャープ パソコンフォーラム'91 展示品特価

### CZ-606DTN

標準価格¥79,800
**台数限定**
**¥57,000**

## アイビット推奨ディスプレイ

シャープ
**CZ-612DGY**
ドットピッチ0.31
チルト台付
**特価¥80,000**

CZ-880D/860Dの代品
シャープ
**CU-14TV**
ドットピッチ0.31
**¥64,800**

シャープ
**CZ-602D-BK**
(15型アナログTV/3モードオートスキャン)
**特価¥75,000**

**FMTV-154** ¥129,200→**¥75,000**
15型デュアルスキャン15K/アナログ21PTVチューナー付

**FMD-PC231D** ¥89,800→**¥45,000**
15型デュアルスキャン15K/24Kアナログ21P

※富士通、NEC、シャープ周辺機器(拡張機器全機種、プリンター他)も常時取り扱っております。

## 富士通 FM TOWNS お買得セット

### FM TOWNS TOWNSモデル20F基本拡張セット

TOWNS20F(本体)
FMT-DP533(カラーディスプレイ)
FMT-KB101(キーボード)
FMT TU-101テレビチューナー
MS DOS(V3.1.L23)
FM秘書
特価セット **¥315,000**

### FM TOWNS TOWNSモデル-1S

TOWNS1-S(本体)
FMT-DP533(カラーディスプレイ)
FMT-KB101(キーボード)
B-276A010システムソフト
特価セット **¥185,000**

FM TOWNSモデル1S 本体特価**¥95,000**

40H、80H等、その他の組合せもご相談下さい。

〈全商品新品完全保証付〉 シャープ、カシオポケコン全機種取扱い。カタログ、価格表ご請求には、72円を添えてお願い致します。

**☎0426-45-3002**(京王線北野駅前店)−3001(本店)−3003(教室)
**FAX.0426-44-6002**

●営業時間/10:00~19:00●電話受付/20:00迄可●定休日/水曜日

**SHARP SUPER XEX SHOP**

アイビット電子株式会社 〒192 東京都八王子市北野町560-5

上記の広告商品はすべて店頭販売もしております。

**全国通信販売** 北海道から沖縄まで

★送料はご注文の際にお問い合わせ下さい。
★掲載の商品は、すべて新品、保証書付きです。
★掲載の商品は充分用意してありますが、ご注文の際は、在庫の確認の上、現金書留または、銀行振込でお申し込み下さい。全商品クレジットでも扱っております。
★お申し込みの際は必ず電話番号を明記して下さい。
★商品、品切れの節はご容赦下さい。

富士銀行八王子支店 (普)1752505

●本誌発売時には上記価格よりさらにお求めやすい価格に変更されている場合があります。●この広告の商品にはすべて送料・消費税は含まれておりません。

ACCESS

# X1エミュレータ

好評発売中

定価¥9,800

X1エミュレータはX68000上でX1シリーズのアプリケーションを実行するためのソフトエミュレータです。X1のアプリケーションを完全にソフトウェアのみでエミュレートしているため、X1上での実行速度と比較して、平均3〜5倍程度おそくなりますが、X68000のマシン上に実現した仮想X1マシンを楽しめます。また、X1とX68000の相互間でファイルを転送するためのユーティリティと専用ケーブルが付属しますので、X1上で作り上げたソフトの資産をX68000上に移行することも簡単にできます。

## X1エミュレータの機能

- X1エミュレータはX1に相当する機能をエミュレート。この仮想コンピュータには最大4つのドライブが仮想的に接続。
- X1エミュレータからみたドライブはHuman68kのドライブ上にあるファイルで仮想的に実現。このファイルはX1用の5″2Dディスクのイメージをファイル転送ユーティリティでまるごと転送したもの。
- X1エミュレータで仮想的に実現したX1は仮想ドライブから起動。このため仮想ドライブ用ファイルには、X1を立ち上げるために必要なHuBASICやCP/Mなどのシステムプログラムが必要。
- X1エミュレータでは、X1の持つVRAMを含むメモリイメージやZ80CPUを仮想的にソフトウェアで実現。

## ファイル転送ユーティリティ

### ディスク転送

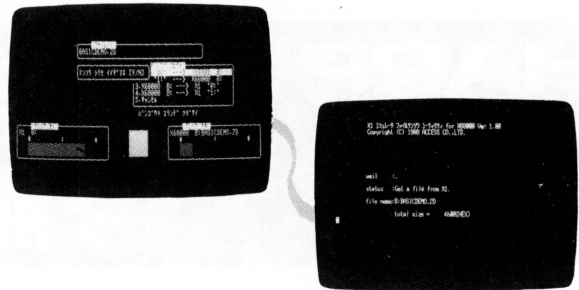
X1ディスク⟷X68000 Human68k(5″2Dディスクイメージファイル)

- X1エミュレータではHuman68k上のディスクイメージファイルを仮想ドライブとして使用。

### ファイル転送

X1 BASIC:CP/M⟷X68000 Human68k

- X1で作ったプログラム&データをX68000上で使用。

※付属の専用ケーブルをX1とX68000に接続してファイルを転送します。

## X1エミュレータQ&A

**Q.** ファイル転送のために別途RS-232Cケーブルを買わないといけないのですか?

**A.** 専用のケーブルが付属しますのでその必要はありません。

**Q.** X1BASICのプログラムをX68000上のX-BASICで使えますか?

**A.** 通常のセーブではコードが違うので使用できませんが、アスキーセーブしたファイルであればX-BASIC上でそのままロード可能です。

**Q.** TurboBASICで作成した住所録などの漢字を含んだデータがあるのですがX68000上にファイル転送できますか?

**A.** X1TurboもX68000も漢字はシフトJISコードなのでファイルの転送は可能です。ただし、漢字ROMを必要とするものはサポートしていません。

**Q.** Turbo用のソフトは動きますか?

**A.** X1用のみでTurbo専用のソフトは動きません。

**Q.** ゲームは動きますか?

**A.** 純粋にBASICでかかれたものは動きますが、プロテクトがかかったものや直接ハードをアクセスするような市販のゲームは動きません。

*タイミング等ハードウェアに依存するようなソフトは、原理上実行できない、もしくは正常に動作しない場合がありますのでご注意ください。
*一部サポートしていない機能があります。

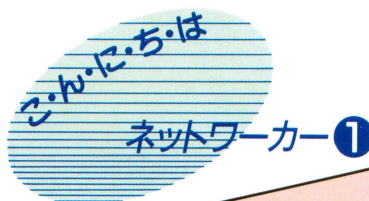
**X1エミュレータ通信販売** 購入希望として住所、氏名、電話番号をお知らせください。注文書をお送り致します。

*この商品価格には消費税は含まれておりません。
*CP/Mはデジタルリサーチ社の商標です。
文中のソフトウェアは各社の商標です。
*製品の仕様、名称は予告なく変更する場合もございますのであらかじめご了承ください。

有限会社アクセス 〒101 東京都千代田区神田神保町1-64 神保町協和ビル7F
☎03(3233)0200(代) FAX.03(3291)7019

パソコン/ワープロ通信ネットワークサービス

# J&P HOTLINE

こ・ん・に・ち・は ネットワーカー①

J&P HOTLINEは私に"パソコン通信"を教えてくれた

## 森田 謙一さん 30歳
（JH161257 もりちゃん）

ソフトウェア開発者

この雑誌広告のための登場者(!?)募集に即、応募してくれた積極派。ソフトウェア開発者になろうとデューダ(?)したのが6年前。それとほぼ時期を同じくしてJ＆P HOTLINEの会員に。昼の休憩時間ならということで会社に取材に伺うと、初対面とは思えない親しみのある笑顔で迎えられました。

## 横浜と大阪を結ぶ社内ネットワークの構築を目指して

森田さんが勤務する会社では、今、オリジナルソフトウェア&システム開発のためにいくつもの分科会が活発に活動中。その一つである社内ネットワークの分科会メンバーである森田さん。横浜の本社と関西支社を合わせて500数十名という大所帯の社内コミュニケーションの充実と、より本格的な社内ネットワークシステムの構築を目指し奮闘中です。

その活動の一環として社内ＢＢＳを試験稼働させ、SysOpとして頑張っているのが森田さん。この社内ＢＢＳの手本となったのはＪ＆Ｐ HOTLINEです。社内ＢＢＳ開設前には、HOTLINE事務局の協力も得て、CUG利用で社内ネットを構築している人からメールをもらって意見や体験談を聞き、「企業内コミュニケーションにおける商用ネットの利用価値についての考察」という論文でみごと社内技術論文賞を受賞。またHOTLINEのデータベースにある「週間クリッピングニュース」は定期的にダウンロードして社内活動に役立てるなどHOTLINEは頼りになる存在です。

CIJネット メインメニュー
[A] 自動ダウンロード　[N] 新アーティクル探索
[B] 電子掲示板　[P] PDSコーナー
[#] チャットルーム　[Q] 質問コーナー
[D] CIJ情報　[#] 電報機能
[E] 終了　[U] 端末環境変更機能
[H] お知らせ　[W] アクセス状況通知機能
[L] アンケートコーナー　[#] 全シグ探索
[M] 電子メール　[Z] ハムレットゲーム
# のサービスは現在行なっていません

**ビジネスシーンだけでなく、森田さんとHOTLINEのお付き合いは多面的。**

★プログラムライブラリー・PCユーザーズクラブ・パソコン256倍活用講座等のＳＩＧでは主にプログラム開発用のＰＤＳダウンロード。
★最近興味を持ち出した「釣り」は、SIGに頻繁に出入りして情報交換。
★直接的にはパソコン通信に関係のない小説・イラストの同人サークルの会員募集をＢＢＳで。めでたく２名の新会員を迎え、オフラインで会合も。

**J&P HOT LINEへのご入会はスタータキットで。**

買ったその日から 2週間無料で アクセスできます。

お求めは、下記のお店へ。又は現金書留にて、¥3,000＋¥90(消費税3%)＝¥3,090を事務局までお送り下さい。
すぐにスタータキットをお送りします。

お問い合わせは　〒556 大阪市浪速区日本橋西1-6-5 上新電機株式会社
J&P HOTLINE事務局宛　TEL.(06)632-2521

**スタータキットのお求めはJ&P各店でどうぞ。**

| 店名 | 住所 | 電話 |
|---|---|---|
| 渋谷店 | 東京都渋谷区道玄坂2丁目28番4号 | ☎(03)3496-4141 |
| 町田店 | 東京都町田市森野1丁目39番16号 | ☎(0427)23-1313 |
| 八王子店 | 東京都八王子市旭町1番1号八王子そごう7F | ☎(0426)26-4141 |
| 立川店 | 東京都立川市幸町4－39－1 | ☎(0425)36-4141 |
| 本厚木店 | 厚木市中町3－4－3 | ☎(0462)25-1548 |
| 富山店 | 富山市桜町2－1－10 | ☎(0764)32-3133 |
| 金沢店 | 金沢市入江2－63 | ☎(0762)91-1130 |
| 寺地店 | 金沢市寺地2－3 | ☎(0762)47-2524 |
| 大須店 | 名古屋市中区大須4丁目2-48 | ☎(052)262-1141 |
| テクノランド | 大阪市浪速区日本橋5丁目6番7号 | ☎(06)634-1211 |
| メディアランド | 大阪市浪速区日本橋5丁目8番26号 | ☎(06)634-1511 |
| ㊗新コスモランド | 大阪市浪速区難波中2丁目1番17号 | ☎(06)634-3111 |
| U. S. LAND | 大阪市浪速区日本橋4丁目9番15号 | ☎(06)634-1411 |
| ビジネスランド | 大阪市北区梅田1-1-3大阪駅前第3ビルB2 | ☎(06)348-1881 |
| 梅田店 | 大阪市北区小松原町1－10 | ☎(06)362-1141 |
| 高槻店 | 高槻市高槻町11番16号 | ☎(0726)85-1212 |
| くずは店 | 枚方市楠葉花園町15番2号 | ☎(0720)56-8181 |
| 千里中央店 | 豊中市新千里東町1-3 SENCHU PAL 2番街4F | ☎(06)834-4141 |
| 摂津富田店 | 高槻市大畑町24－10 | ☎(0726)93-7521 |
| 寝屋川店 | 寝屋川市緑町4－20 | ☎(0720)34-1166 |
| 藤井寺店 | 藤井寺市岡2丁目1番33号 | ☎(0729)38-2111 |
| 岸和田店 | 岸和田市土生町2451－3 | ☎(0724)37-1021 |
| さんのみや1ばん館 | 神戸市中央区八幡通3-2-16 | ☎(078)231-2111 |
| 西宮店 | 兵庫県西宮市河原町5－11 | ☎(0798)71-1171 |
| 姫路店 | 姫路市東延末1丁目1番住友生命姫路南ビル1F | ☎(0792)22-1221 |
| 京都寺町店 | 京都市下京区寺町通仏光寺下ル恵比須之町549 | ☎(075)341-3571 |
| 京都近鉄店 | 京都市下京区烏丸通七条下ル東塩小路町702 | ☎(075)341-5769 |
| 和歌山店 | 和歌山市元寺町4丁目4番地 | ☎(0734)28-1441 |
| 奈良1ばん館 | 奈良市三条町478－1 | ☎(0742)27-1111 |
| 郡山インター店 | 大和郡山市横田693－1 | ☎(07435)9-2221 |
| 熊本店 | 熊本市手取本町4－12 | ☎(096)359-7800 |

7月号 1991年7月1日発行（毎月1回1日発行）通巻111号 昭和58年11月2日第三種郵便物認可 ソフトバンク株式会社発行 Printed in Japan 特別定価600円（本体583円）

T4910217907602 雑誌 02179-7